AFOSR-TR-97 0686

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 31, 1997	FINAL TECHNICAL REPORT 1 Jun 96 to 30 Nov 96

4. TITLE AND SUBTITLE
AN OPEN COMPUTATIONAL TOOL FOR ANALYTICAL DESIGN OF MULTI-STEP FORGING PROCESSES

5. FUNDING NUMBERS
F49620-96-C-0028

6. AUTHOR(S)
GAL BERKOOZ, ESTABAN MARIN

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
BEAM TECHNOLOGIES, INC.
110 N. CAYUGA STREET
ITHACA, NY 14850-4331

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
110 DUNCAN AVENUE SUITE B115
BOLLING AFB DC 20332-8050

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
F49620-96-C-0028

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

In this work we have developed (i) a preliminary methodology for optimizing multi-step forging process and (ii) the corresponding computational tool using or computing substrate PDESolve. The methodology is based on the solution of the direct and sensitivity problems coupled to an optimization algorithm and is applicable for general optimization-based design problems in mechanics. In particular, it can be used for the design of preform shape and die geometry in forging processes. Our computing substrate PDESolve is an open environment programming tool based on the object oriented languageC++ and has specific features that allow the user to program a problemat a high level. The methodology has been implemented using PDEsolve and will be applied to optimize the preform shape in a one-step open die forging operation (upsetting). the solution of this example as well as others in mechanics show that PDESolve can be used as an effective numerical tool for solving optimization-based design problems in engineering.

19971204 167

14. SUBJ.

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

An Open Computational Tool for Analytical Design of Multi-Step Forging Processes

Contract F49620-96-C-0028

0003AA - **Final Technical Report**

Gal Berkooz, Esteban Marin
BEAM Technologies, Inc.
110 N. Cayuga St.
Ithaca, NY 14850-4331

Prepared for USAF, Air Force Office of Scientific Resarch
Bolling AFB, D.C.

March 31, 1997

Summary

In this work we have developed (i) a preliminary methodology for optimizing multi-step forging process and (ii) the corresponding computational tool using our computing substrate PDESolve. The methodology is based on the solution of the direct and sensitivity problems coupled to an optimization algorithm and is applicable for general optimization-based design problems in mechanics. In particular, it can be used for the design of preform shape and die geometry in forging processes. Our computing substrate PDESolve is an open environment programming tool based on the object oriented language C++ and has specific features that allow the user to program a problem at a high level. The methodology has been implemented using PDESolve and will be applied to optimize the preform shape in a one-step open die forging operation (upsetting). The solution of this example as well as others in mechanics show that PDESolve can be used as an effective numerical tool for solving optimization-based design problems in engineering.

1 Introduction

Forging is a primary deformation process that plays an important role in the production of net-shape or near net-shape components for aerospace structural applications. During metal forging, large plastic strains are imparted to the workpiece. These strains cause microstructural changes that affect the material response during the forging process itself as well as its final mechanical properties. High strains may affect the workability of the workpiece due to production of voids and fractures, and the appearance of shear bands (flow localization), all of which will result in a defective product. On the other hand, material properties such as toughness and strength, as well as the formability of the forged part during secondary processing and machining are very sensitive to the developed microstructure. Moreover, the nonuniform distribution of plastic strains, which is typically produced by most deformation processes, may induce undesirable residual stresses to the material which will affect its performance in service.

Current basic research at the Air Force is emphasizing the development of fundamental material and process models to address many of these aspects so as to improve the workability of the material as well as the mechanical properties of the formed part. Forging as well as other forming processes (rolling, extrusion, sheet forming) are being investigated for a number of materials of specific interest to the Air Force, such as various conventional titanium and nickel-based alloys. Specific aspects of the research focus on thermomechanical process modeling, microstructure evolution modeling (grain size, grain shape, texture), and structure/property characterization (constitutive laws).

Because the structural performance of aerospace systems can be enhanced by a judicious tailoring of their shape and microstructure, optimization procedures are becoming an important part of the design process. In particular, during forging the final geometry and state (properties) of a forged part depend on a number of processing conditions, such as forging temperature, initial workpiece geometry (preform shape), die geometry, lubrication conditions, loading conditions (ram velocity), as well as the type of material being forged. Designing a forging process to meet specific requirements of performance for the final product then requires the control/optimization of these processing variables.

A number of works have addressed the optimum design of the forging process. Kobayashi *et al.* [21] introduce the so-called backward tracing technique for preform design. Han *et al.* [17] improved on this method by including design optimization strategies (sensitivity analysis and specific objective functions) and solved the design of intermediate forging die shapes. Both of these methods start from the final product shape and trace back the loading path of the forging process to determine the required shape of the preform and intermediate dies. Recently, sensitivity analysis and optimization techniques have been coupled with the solution of the direct (forward) problem to solve the preform and die shape design problems in forging. Applications of this approach to solve preform and die design problems are given by Badrinarayanan & Zabaras [2, 3] for one-step forging and by Fourment *et al.* [10] for two-step forging. The numerical tool used in all these studies was the finite element method.

The work in Phase I, which will be continued in Phase II, is aimed at developing an optimization-based forging design approach to determine the processing variables that give the best possible compatibility between final product shape requirement and desired material state. In particular, this work has built on previous studies [2, 3] to address the design optimization of preform shape and die geometry. Specifically, in this Phase we have mainly focussed on

- The development of an optimization-based design methodology for multi-step forging, and
- The implementation of the methodology using our computational substrate PDESOLVE for solving 2-D forging design problems.

In essence, the methodology is based on the solution of the direct and sensitivity problems defining the forging process coupled to a function optimizer. Because the present version of PDESOLVE is based on the finite difference method, the governing equations for the direct and sensitivity problems have been implemented in their differential (strong) form. Currently, a PDESOLVE version using the finite element method (which uses the variational or weak form of the equations) is being developed and will be used as the numerical tool for Phase II. As an application example of the methodology, the PDESOLVE code has been used to solve the preform shape design for open die forging (upsetting), modeled as a 2-D problem. This application uses a hypoelastic-viscoplastic model for the material description and assumes sticking friction at the die-workpiece interface.

The organization of the report is as follows. Section 2 presents a brief literature review on sensitivity analysis as applied to solid mechanics problems. Section 3 details the methodology for optimization-based design of multi-step forging together with the formulation of the equations (in strong form) for solving the direct and the sensitivity problems in nonlinear solid mechanics. Section 4 presents the main features of our computational substrate PDESOLVE and gives some application problems of sensitivity analysis and design optimization in solid mechanics. Here, specific examples in linear elasticity and in small deformation plasticity are solved. Section 5 gives an application example of the methodology using PDESOLVE to the preform design of one-step open die forging (upsetting). Section 6 presents a road map for an extended constitutive framework to be included in Phase II and Section 7 gives the conclusions of the study.

In this report, we use direct tensor notation [14] to develop the governing equations. For this purpose, vectors and second order tensors are indicated with bold face lower and upper case letters, respectively, e.g. $\mathbf{a}$ and $\mathbf{A}$; and fourth order tensors are denoted by calligraphic bold letters, e.g. $\boldsymbol{\mathcal{C}}$. The dyadic product of two vectors $\mathbf{a}$ and $\mathbf{b}$ is indicated as $\mathbf{a} \otimes \mathbf{b}$. Tensor operations between two second order tensors $\mathbf{A}$ and $\mathbf{B}$ are indicated as $\mathbf{A}\cdot\mathbf{B}$ for the inner product (a second order tensor), $\mathbf{A} \otimes \mathbf{B}$ for the dyadic product (a fourth order tensor), and $\mathbf{A}:\mathbf{B}$ for the scalar product (a scalar). Contraction operations over two indices between a fourth $\boldsymbol{\mathcal{C}}$ and second $\mathbf{A}$ order tensors are represented as $\boldsymbol{\mathcal{C}}:\mathbf{A}$ (a second order tensor).

2 Sensitivity Analysis: The Basis for Optimization-Based Design in Solid Mechanics

Sensitivity analysis refers to methods for computing the rate of change of response quantities, such as displacements and stresses, with respect to changes of the design parameters. This first derivative information (design sensitivity coefficients) is used in gradient-based design optimization algorithms to compute the gradient of the objective function and constraints with respect to the design variables. Typical design variables in solids mechanics include shape (geometrical) parameters, sizing parameters, material parameters, and boundary condition parameters.

Basically, three different approaches have been used for computing design sensitivity coefficients [18]: The finite difference method (FDM), the adjoint structure method (ASM) and the direct differentiation method (DDM). In the FDM, multiple solutions of the direct problem are carried out with different values of a single design parameter and the solutions obtained are used to calculate the sensitivities. The number of evaluations required depends on the order of the finite difference representation used. The main drawback of this method is the critical dependence of the calculated sensitivity upon the step size of the design variable. Because this method is costly and can introduce large errors, it is seldom used in practice.

The ASM is based on formulating an auxiliary (adjoint) boundary value problem for the sensitivity coefficients. This problem is constructed by imposing certain initial stresses or strains and a set of boundary conditions defined according to the constraint functional (objective function) whose design sensitivity is desired. The solution of the adjoint structure permits then the explicit evaluation of the required sensitivities for the direct problem. This approach is very effective for systems defined by linear field equations, such as linear elasticity and structural mechanics [18]. However, it is not suitable for history dependent problems since the adjoint system can only be solved after the analysis of the direct problem has been completed [30]. Computationally, this means that the data calculated during the direct analysis need to be either saved or recalculated for sensitivity analysis. This is quite inefficient for computer implementation. In the DDM, on the other hand, the sensitivity equations are derived by taking the variations of the continuum field equations of the direct problem (expressed in either differential or variational form) with respect to small changes of the design parameters. The resulting equations are linear in the sensitivity of the response variables. A special feature of this method is that, for history dependent problems, the sensitivity equations can be solved once the solution of the direct problem for a particular load step is obtained [30]. Hence, the direct and sensitivity problems can be solved in parallel, making the computer implementation of the method quite straightforward. For this reason, the DDM is recommended for highly nonlinear problems with a path dependence response [30], such as the large deformation of inelastic solids.

Sensitivity analysis and optimization for linear problems in mechanics (linear elasticity, structural mechanics) are well established now. A number of excellent books [16, 18, 28] have been published that gives a rigorous mathematical treatment of sensitivity analysis for linear systems [28] as well as its application to solve engineering problems [16, 18]. In this area, the above three approaches for computing sensitivities together

with optimization techniques have been widely applied for engineering design using the finite element method and the boundary element method. Particular applications are in the automobile and aerospace industries. The reader is referred to a recent comprehensive survey paper [8] for details.

Sensitivity analysis and optimization techniques as applied to nonlinear mechanics have recently attracted the attention of researches. Quite a few publications deal with the derivation of sensitivity equations for problems involving geometric and material nonlinearities in structures and continua [15, 20, 25, 30, 31, 32, 33]. Although both the ASM and the DDM have been used in most of these studies [20, 30], the DDM is typically recommended and used for computer implementation with the finite element method [1] and the boundary element method [25]. Both total [30] and updated Lagrangian [20] descriptions have been presented. Of particular interest for the present work is the study reported in [2, 3] which deals with the application of sensitivity analysis and optimization techniques to the design of preform shape and die geometries for one-step forging processes. This study will be used as a guide for the present development.

In the present work, we mainly focus on shape design sensitivity analysis using the DDM for nonlinear solid mechanics problems. Our particular interest is in modeling the finite deformation of an elasto-viscoplastic material as a means to analyze metal forging processes. In this work, we will couple the solution of the direct and sensitivity problems with an optimizer to develop an optimization-based design approach of multi-step forging processes.

3 A General Framework For Optimization-Based Design Of Forming Processes: Multi-Step Forging

3.1 General Approach for Optimization-Based Forging Design

The methodology for solving general optimization-based design problems requires the solution of the governing equations for the direct and sensitivity problems, coupled to a function optimization algorithm. For the specific case of the forging process, the developed methodology has the following features:

1. The direct problem is formulated as the large-deformation analysis of an elasto-viscoplastic material. Under isothermal and quasi-static conditions, this deformation process is governed by the equilibrium equations, the kinematics equations, the constitutive relations and appropriate initial and boundary conditions. These non-linear differential equations are written using an updated Lagrangian description and solved numerically using an incremental strategy.

2. The sensitivity problem is formulated using the DDM [30]. In this method, the sensitivity equations are derived by considering the variations of the governing equations of the direct problem with respect to small changes in the process parameters (design variables). As mentioned before, this method has proven adequate for highly-nonlinear problems with a path-dependent material response [30]. The sensitivity equations, which are linear, are written using a total Lagrangian description. Their numerical solution is also carried out incrementally and in parallel to the solution of the direct problem.

3. The minimization of the objective function is carried out using standard optimization procedures based on either a gradient-based method or a quasi-Newton-based method [12].

In this work (Phase I), we focus on preform (initial workpiece) and die shape design problems. In this context, a typical shape optimization problem can be stated as:

$$\begin{aligned} \min \quad & \phi(\boldsymbol{\beta}), \quad \boldsymbol{\beta} \in R^n && (1)\\ \text{subject to} \quad & h_j(\boldsymbol{\beta}) = 0, \quad \text{for } j = 1, ..., m && (2)\\ & g_k(\boldsymbol{\beta}) \geq 0, \quad \text{for } k = 1, ..., p && (3) \end{aligned}$$

where $\phi(\boldsymbol{\beta})$ is the objective function, $\boldsymbol{\beta}$ is the shape design variable vector with n components, and $h_j(\boldsymbol{\beta})$ and $g_k(\boldsymbol{\beta})$ are equality and inequality constraints, respectively. A general form for the objective and constraints functions in non-linear mechanics can be represented by the performance functional

$$\Phi(\boldsymbol{\beta}) = \int_V f(\mathbf{u}(\boldsymbol{\beta}), \mathbf{E}(\boldsymbol{\beta}), \mathbf{T}(\boldsymbol{\beta}))dV \tag{4}$$

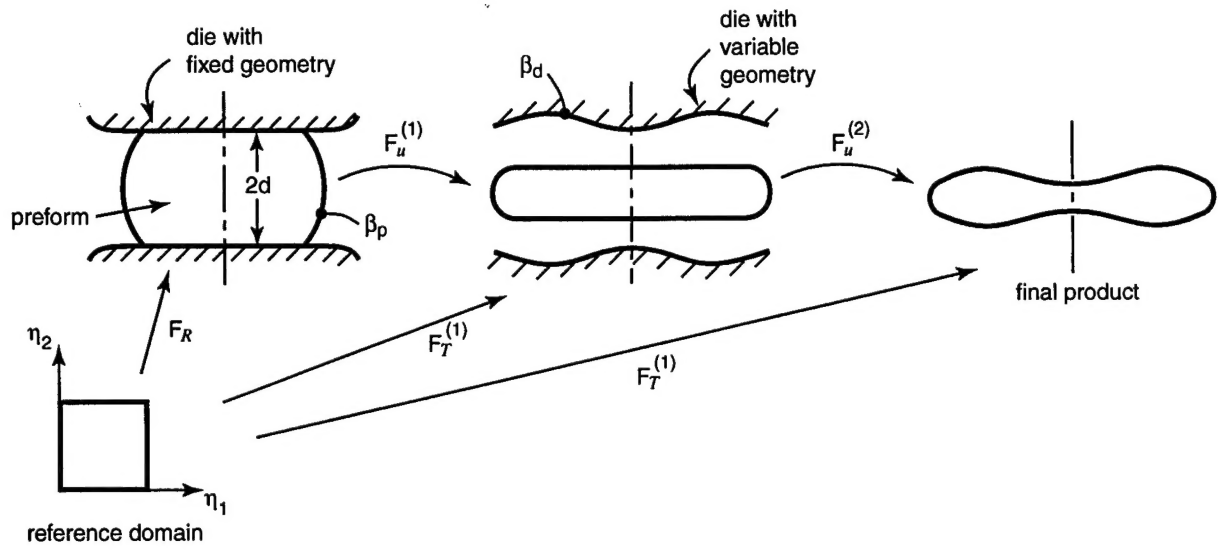

Figure 1: A two-step forging process.

where $\mathbf{u}, \mathbf{E}$ and $\mathbf{T}$ represent displacement, strain and stress tensor fields, respectively, f is a known function of its arguments, and V is the current volume of the body. Here, $\mathbf{E}$ and $\mathbf{T}$ represent some general strain and stress measures, respectively.

Standard optimization techniques require information about the objective function and its derivatives. In the particular case of shape design optimization, the derivatives of ϕ are computed by taking the shape design variations of ϕ with respect to the design variables β. Using the notation $\delta(\cdot) = \frac{\partial(\cdot)}{\partial \beta}\delta\beta$ to represent the variation (sensitivity) of the quantity $(\cdot)$ with respect to β, the design variation $\delta\Phi$ can be written as

$$\delta\Phi = \int_V (\delta f dV + f\delta dV) \tag{5}$$

where

$$\delta f = \frac{\partial f}{\partial \mathbf{u}}\delta\mathbf{u} + \frac{\partial f}{\partial \mathbf{E}}\delta\mathbf{E} + \frac{\partial f}{\partial \mathbf{T}}\delta\mathbf{T} \tag{6}$$

$$\delta dV = tr(\delta\mathbf{L})dV \quad , \quad \delta\mathbf{L} = \frac{\partial \delta\mathbf{x}}{\partial \mathbf{x}} \tag{7}$$

Here, $\mathbf{x}$ is the position vector of a particular material point in the body, and $\delta\mathbf{L}$ is the variation of the design velocity gradient [3, 9]. The design velocity represents the rate of change of the position vector $\mathbf{x}$ due to changes in the design parameters, i.e., $\mathbf{v} = \frac{\partial \mathbf{x}}{\partial \beta}$. Evaluating the functions Φ and $\delta\Phi$ requires the solution of the governing equations for the direct and sensitivity problems to compute the quantities $(\mathbf{u}, \mathbf{E}, \mathbf{T})$ and their corresponding sensitivities $(\delta\mathbf{u}, \delta\mathbf{E}, \delta\mathbf{T})$, respectively.

For multi-step forging, the process design is formulated as the design of the preform and the design of the dies at each stage of the forging operation. To illustrate the procedure to be followed in solving the multi-step forging optimization problem, we will consider the two-step forging shown in fig. 1 modeled as a 2-D problem. Here, the dies are assumed to be rigid. In this problem, we want to optimize the geometry of the preform and the intermediate dies in order to get a product with a specified geometry. To simplify the explanation, we will assume that the geometry of the initial die is fixed. The particular objective function for this optimization problem can be written as

$$\Phi = \int_V (\Omega(\beta_p, \beta_d) - \bar{\Omega})^2 dV \tag{8}$$

where Ω is the computed shape from process simulation and is an implicit function of the response field variables, $\bar{\Omega}$ is the desired shape, and β_p and β_d are the design parameters defining the preform and final die geometry, respectively. Using a parametric representation for these shapes such as Bezier curves, the design parameters can be identified as the control points of the curves. The optimization algorithm will then require

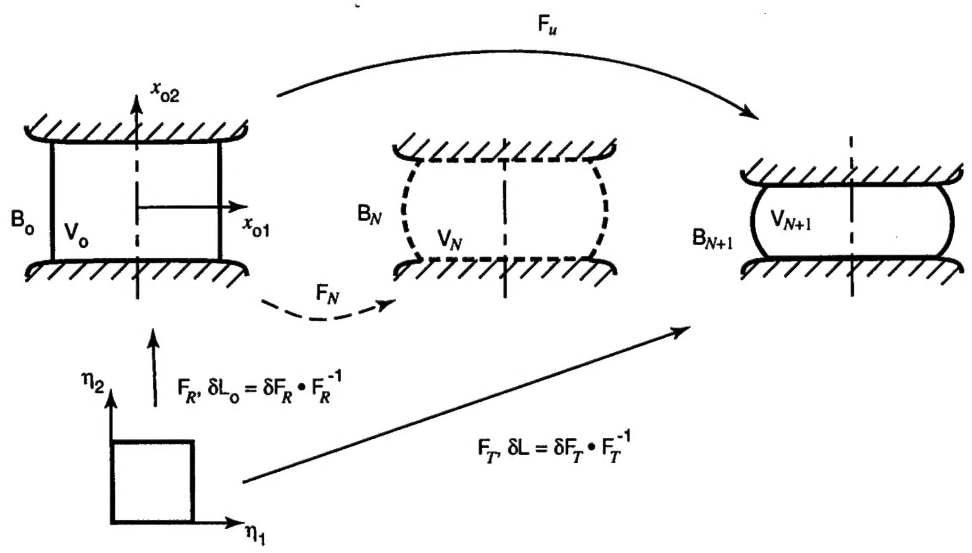

Figure 2: A general step of a forging process.

information about the function Φ and its variation (sensitivity) with respect to the design parameters β_p and β_d, i.e.,

$$\delta_{\beta_p}\Phi = \frac{\partial\Phi}{\partial\beta_p}\delta\beta_p \quad , \quad \delta_{\beta_d}\Phi = \frac{\partial\Phi}{\partial\beta_d}\delta\beta_d \tag{9}$$

To compute these quantities, we need to solve the direct and sensitivity problems that describe the large strain inelastic deformation of a body. The body's deformation will be described using the updated Lagrangian scheme for the direct problem and the total Lagrangian scheme for the sensitivity problem.

In what follows, we detail the framework for the analysis of the direct and sensitivity problems in finite-deformation elasto-viscoplasticity. Then, we present the methodology for the design of the two-step forging process presented in fig. 1. We note here that the governing equations for the direct and sensitivity problems will be presented in their differential form, which is suitable for the finite difference method (Phase I). The variational formulation of these equations to be used with the finite element method (Phase II) is given in the appendix.

3.2 Formulation of the Direct Problem

The direct problem consists of finding the deformation history and material state of a body subjected to the action of known external forces. In large deformation problems, such as forging, the configuration of the body changes due to loading. Figure 2 shows a general step of a forging process. Here, we denote the configurations of the body in the undeformed state and in the deformed state at times t_N and t_{N+1} as $\mathcal{B}_0$, $\mathcal{B}_N$ and $\mathcal{B}_{N+1}$, respectively. We assume that the configurations $\mathcal{B}_0$ and $\mathcal{B}_N$ are known and that configuration $\mathcal{B}_{N+1}$ needs to be determined. A particular material point in these configurations will be located with respect to a fixed spatial reference frame by the vector positions $\mathbf{x}_0, \mathbf{x}_N, \mathbf{x}_{N+1}$, respectively.

As mentioned above, the direct problem is formulated using the updated Lagrangian scheme. Hence, the equilibrium equations and boundary conditions to be solved for determining the unknown configuration $\mathcal{B}_{N+1}$ are written relative to the known configuration $\mathcal{B}_N$ as

$$\nabla_N \cdot \mathbf{P} + \rho_N \mathbf{b} = 0 \qquad \text{in } V_N \tag{10}$$

$$\mathbf{P} \cdot \mathbf{n}_N = \bar{\mathbf{t}}_N \qquad \text{on } A_N^{\sigma} \tag{11}$$

$$\mathbf{u} = \bar{\mathbf{u}} \qquad \text{on } A_N^{\mathbf{u}} \tag{12}$$

where $\nabla_N = \frac{\partial}{\partial \mathbf{x}_N}$, $\mathbf{P}$ is the first Piola-Kirchoff stress in configuration $\mathcal{B}_{N+1}$ as referred to the configuration $\mathcal{B}_N$ ($\mathbf{P} = \mathbf{P}_N^{N+1}$), $\mathbf{b}$ is the body force per unit mass, ρ_N is the mass density in configuration $\mathcal{B}_N$, $\mathbf{n}_N$ is the unit normal to the surface A_N^{σ}, $\mathbf{u}$ is the displacement field, and $\bar{\mathbf{t}}_N$ and $\bar{\mathbf{u}}$ are prescribed tractions and displacements

on the surfaces A_N^σ and A_N^u, respectively. The first Piola-Kirchhoff stress is expressed as (following Gurtin's convention [14])

$$\mathbf{P} = det(\mathbf{F})\boldsymbol{\sigma} \cdot \mathbf{F}^{-T} = \mathbf{F} \cdot \mathbf{S} \tag{13}$$

where $\mathbf{F}$ is the relative deformation gradient ($\mathbf{F} = \mathbf{F}_N^{N+1}$), $\boldsymbol{\sigma}$ is the Cauchy stress and $\mathbf{S}$ is the second Piola-Kirchhoff stress in configuration $\mathcal{B}_{N+1}$ as referred to the configuration $\mathcal{B}_N$ ($\mathbf{S} = \mathbf{S}_N^{N+1}$). We note here that the contact between the die and the workpiece during forging is introduced in eqs.(10)-(12) as a traction-type boundary condition.

The elasto-viscoplastic constitutive material response can be modeled using a phenomenologically-based (continuum plasticity [23] or micromechanically-based (polycrystal plasticity [5]) approach. In the present work, we use models based on the first approach. In this respect, we use the following continuum hypoelastic-viscoplastic description [22]

$$\overset{\nabla}{\boldsymbol{\sigma}} = \boldsymbol{\mathcal{C}}:(\mathbf{D} - \mathbf{D}^p) \tag{14}$$

$$\mathbf{D}^p = \sqrt{\frac{3}{2}}\dot{\bar{\epsilon}}^p \mathbf{N} \tag{15}$$

$$\dot{\bar{\epsilon}}^p = f(\bar{\sigma}, s) \tag{16}$$

$$\dot{s} = g(\bar{\sigma}, s) \tag{17}$$

where

$$\overset{\nabla}{\boldsymbol{\sigma}} = \dot{\boldsymbol{\sigma}} - \mathbf{W}\cdot\boldsymbol{\sigma} + \boldsymbol{\sigma}\cdot\mathbf{W} \tag{18}$$

$$\mathbf{N} = \frac{\boldsymbol{\sigma}'}{\| \boldsymbol{\sigma}' \|} \quad , \quad \bar{\sigma} = \sqrt{\frac{3}{2}} \| \boldsymbol{\sigma}' \| \tag{19}$$

with $\boldsymbol{\sigma}' = \boldsymbol{\sigma} - \frac{1}{3}tr(\boldsymbol{\sigma})\mathbf{I}$. In these equations, $\mathbf{D}$ and $\mathbf{D}^p$ are the rate of deformation and its plastic part, respectively, $\mathbf{W}$ is the continuum spin, $\dot{\bar{\epsilon}}^p$ is the effective plastic strain rate, $\bar{\sigma}$ is the effective stress, s is the material hardness and $\boldsymbol{\mathcal{C}}$ is the constant elasticity tensor given by $\boldsymbol{\mathcal{C}} = 2\mu\mathcal{I} + \lambda\mathbf{I}\otimes\mathbf{I}$. Here, $\mathcal{I}$ and $\mathbf{I}$ represent the fourth-order and second-order identity tensors, and μ, λ are Lame's constants. Also, $\| \boldsymbol{\sigma}' \| = (\boldsymbol{\sigma}':\boldsymbol{\sigma}')^{1/2}$.

From a numerical point of view, the time integration of the constitutive equations is carried out using an objective integration scheme [34] . In this procedure, the constitutive equations (14)-(17) are transformed (pull back) using the rotation tensor $\mathbf{R}$ obtained from $\mathbf{F} = \mathbf{R}\cdot\mathbf{U}$, where $\mathbf{U}$ is the right Cauchy-Green strain tensor. The transformed constitutive equations are [22]

$$\dot{\bar{\boldsymbol{\sigma}}} = \boldsymbol{\mathcal{C}}:(\bar{\mathbf{D}} - \bar{\mathbf{D}}^p) \tag{20}$$

$$\bar{\mathbf{D}}^p = \sqrt{\frac{3}{2}}\dot{\bar{\epsilon}}^p \bar{\mathbf{N}} \tag{21}$$

$$\dot{\bar{\epsilon}}^p = f(\bar{\sigma}, s) \tag{22}$$

$$\dot{s} = g(\bar{\sigma}, s) \tag{23}$$

where $\dot{\bar{\boldsymbol{\sigma}}} = \mathbf{R}^T\cdot\overset{\nabla}{\boldsymbol{\sigma}}\cdot\mathbf{R}$, $\bar{\mathbf{D}} = \mathbf{R}^\mathbf{T}\cdot\mathbf{D}\cdot\mathbf{R}$, and similarly for $\bar{\mathbf{D}}^\mathbf{p}$ and $\bar{\mathbf{N}}$. The equations for the rotated-neutralized tensors $\bar{\boldsymbol{\sigma}}$, $\bar{\mathbf{D}}$, $\bar{\mathbf{D}}^\mathbf{p}$ and $\bar{\mathbf{N}}$, Eqs.(20)-(23), are then integrated using an implicit integration procedure (backward Euler) [22, 34].

The numerical solution of Eqs.(10)-(12) and Eqs.(20)-(23) is based on an incremental, displacement-based strategy. For this purpose, the specified time interval is divided into a certain number of time steps. At each time step, two coupled iterative procedures are defined: the equilibrium iterations and the constitutive iterations.

The equilibrium iterations are carried out to compute the displacement field $\mathbf{u}$. Here, we use the differential form of these equations, written in a suitable form for algorithm development. This form is obtained as follows. Using the updated Lagrangian nature of the formulation, we can write

$$\mathbf{F} = \mathbf{F}_N^{N+1} = \mathbf{F}_N^N + \Delta\mathbf{F} = \mathbf{I} + \Delta\mathbf{F} \tag{24}$$

$$\mathbf{S} = \mathbf{S}_N^{N+1} = \mathbf{S}_N^N + \Delta\mathbf{S} = \boldsymbol{\sigma}_N + \Delta\mathbf{S} \tag{25}$$

Hence, $\mathbf{P}$, eq.(13), can be expressed as

$$\mathbf{P} = \mathbf{P}_N^{N+1} = \boldsymbol{\sigma}_N + \Delta\mathbf{S} + \Delta\mathbf{F}\cdot\boldsymbol{\sigma}_N + \Delta\mathbf{F}\cdot\Delta\mathbf{S} \tag{26}$$

Neglecting the higher order term $\Delta\mathbf{F}\cdot\Delta\mathbf{S}$ in Eq.(26), we can write Eqs(10)-(12) as

$$\begin{aligned}
\boldsymbol{\nabla}_N\cdot(\Delta\mathbf{S}+\Delta\mathbf{F}\cdot\boldsymbol{\sigma}_N) &= -\rho_N\mathbf{b}-\boldsymbol{\nabla}_N\cdot\boldsymbol{\sigma}_N \qquad \text{in } V_N & (27)\\
(\Delta\mathbf{S}+\Delta\mathbf{F}\cdot\boldsymbol{\sigma}_N)\cdot\mathbf{n}_N &= \bar{\mathbf{t}}_N-\boldsymbol{\sigma}_N\cdot\mathbf{n}_N \qquad \text{on } A_N^{\sigma} & (28)\\
\Delta\mathbf{u} &= \Delta\bar{\mathbf{u}} \qquad \text{on } A_N^{\mathrm{u}} & (29)
\end{aligned}$$

Using the definition of the second Piola-Kirchhoff stress $\mathbf{S} = det(\mathbf{F})\mathbf{F}^{-1}\cdot\boldsymbol{\sigma}\cdot\mathbf{F}^{-T}$, the pull forward of the Cauchy stress $\boldsymbol{\sigma} = \mathbf{R}\cdot\bar{\boldsymbol{\sigma}}\cdot\mathbf{R}^T$, the polar decomposition of the relative deformation gradient $\mathbf{F} = \mathbf{R}\cdot\mathbf{U}$, and the updated Lagrangian assumption, one can show that

$$\Delta\mathbf{S} = \Delta\bar{\boldsymbol{\sigma}} - \Delta\mathbf{U}\cdot\boldsymbol{\sigma} - \boldsymbol{\sigma}\cdot\Delta\mathbf{U} + tr(\Delta\mathbf{F})\boldsymbol{\sigma} \tag{30}$$

$$\Delta\mathbf{U} = (\Delta\mathbf{F})_s \quad , \quad \Delta\mathbf{F} = \frac{\partial\Delta\mathbf{u}}{\partial\mathbf{x}_N} \tag{31}$$

where the subscript s denotes the symmetric part of $(\cdot)$. In Eq.(30), the stress increment $\Delta\bar{\boldsymbol{\sigma}}$ is related to the strain increment $\Delta\bar{\mathbf{E}} = \int_{t_N}^{t_{N+1}}\bar{\mathbf{D}}dt = 2(\mathbf{U}-\mathbf{I})\cdot(\mathbf{U}+\mathbf{I})^{-1}$ using the elasto-viscoplastic constitutive matrix $\bar{\boldsymbol{\mathcal{L}}}$ (consistent tangent operator [27])

$$\Delta\bar{\boldsymbol{\sigma}} = \bar{\boldsymbol{\mathcal{L}}}{:}\Delta(\Delta\bar{\mathbf{E}}) \tag{32}$$

where $\Delta(\Delta\bar{\mathbf{E}}) = (\Delta\mathbf{F})_s = \Delta\mathbf{U}$ and

$$\bar{\boldsymbol{\mathcal{L}}} = 2\hat{\mu}\boldsymbol{\mathcal{I}} + (\kappa - \frac{2}{3}\hat{\mu})\mathbf{I}\otimes\mathbf{I} - 2\mu(\eta - c)\bar{\mathbf{N}}\otimes\bar{\mathbf{N}} \tag{33}$$

$$\hat{\mu} = \eta\mu \quad , \quad \eta = \frac{\bar{\sigma}}{\bar{\sigma}*} \quad , \quad c = \frac{b_2}{a_1 b_2 + a_2 b_1} \tag{34}$$

In these equations, κ is the elastic bulk modulus, $\bar{\sigma}*$ is a trial effective stress (terminology related to the constitutive integration scheme), and the coefficients a_1, a_2, b_1, b_2 depend on the constitutive functions f and g, Eqs.(16)-(17).

Using Eqs.(30)-(34), we can formulate the boundary value problem (27)-(29) in terms of the incremental displacement $\Delta\mathbf{u}$ and then solve the resulting equations using the finite difference method. The current version of our computational substrate PDESOLVE will be used for this purpose.

The constitutive iterations, on the other hand, are carried out to update the Cauchy stress $\boldsymbol{\sigma}$ and the material hardness s. For this purpose, the bar-transformed constitutive equations are integrated using the backward Euler scheme (radial return method). Details of the algorithm can be found elsewhere [22] and will not be presented here. In the context of software development, the constitutive integration procedure will be added to PDESOLVE as a particular C++ class and then coupled to the solution of Eqs.(27)-(29) as an integral part of the solution procedure.

3.3 Formulation of the Sensitivity Problem

The formulation for the sensitivity problem is based on the total Lagrangian scheme. This particular scheme for the sensitivity problem avoids difficulties in distinguishing at the current stage of the process between the initial shape design variation and those due to the deformation process itself. Hence, the undeformed configuration $\mathcal{B}_0$ is taken as the reference one to write the equilibrium equations and the boundary conditions.

In the present work, we use the direct differentiation method to compute the sensitivity of the equilibrium equations and the boundary conditions. As before, the notation $\delta(\cdot)$ will be used to represent the variation (sensitivity) of the quantity $(\cdot)$ with respect to the design variables $\boldsymbol{\beta}_p$ and $\boldsymbol{\beta}_d$. Taking the variation of the equilibrium equations and boundary conditions written in the configuration $\mathcal{B}_{N+1}$ as referred to the configuration $\mathcal{B}_0$, we obtain the following sensitivity equations

$$\begin{aligned}
\boldsymbol{\nabla}_0\cdot\delta\mathbf{P} - \boldsymbol{\nabla}_0\mathbf{P}{:}\delta\mathbf{L}_0^T + \delta\rho_0\mathbf{b} &= 0 \qquad \text{in } V_0 & (35)\\
(\delta\mathbf{P} - \mathbf{P}\cdot\delta\mathbf{L}_0^T)\cdot\mathbf{n}_0 &= \delta\bar{\mathbf{t}}_0 + [(\delta\mathbf{F}_u\cdot\mathbf{F}_u^{-1}){:}(\mathbf{I}-\mathbf{n}\otimes\mathbf{n}) - \\
&\qquad (\mathbf{F}_u\cdot\delta\mathbf{L}_0\cdot\mathbf{F}_u^{-1}){:}(\mathbf{n}\otimes\mathbf{n})]\bar{\mathbf{t}}_0 \qquad \text{on } A_0^{\sigma} & (36)\\
\delta\mathbf{u} &= \delta\bar{\mathbf{u}} \qquad \text{on } A_0^{\mathrm{u}} & (37)
\end{aligned}$$

where $\nabla_0 = \frac{\partial}{\partial \mathbf{x}_0}$, $\mathbf{P}$ is the first Piola-Kirchhoff stress in configuration $\mathcal{B}_{N+1}$ as referred to configuration $\mathcal{B}_0$ ($\mathbf{P} = \mathbf{P}_0^{N+1}$), ρ_0 is the mass density in configuration $\mathcal{B}_0$, $\mathbf{n}_0$ is the unit normal to the surface A_0^σ, $\bar{\mathbf{t}}_0$ is the prescribed traction as referred to configuration $\mathcal{B}_0$ ($\bar{\mathbf{t}}_0 = \frac{dA_0^\sigma}{dA^\sigma}\bar{\mathbf{t}}$, $\bar{\mathbf{t}}$ is the prescribed traction at $\mathcal{B}$), $\bar{\mathbf{u}}$ is the prescribed displacement and $\mathbf{F}_u$ is the total deformation gradient. The unit normal $\mathbf{n}$ in the deformed configuration $\mathcal{B}_{N+1}$ is related to $\mathbf{n}_0$ by $\mathbf{n} = det(\mathbf{F}_u)\frac{dA_0^\sigma}{dA^\sigma}\mathbf{F}_u^{-T} \cdot \mathbf{n}_0$. Also, in these equations, $\delta\mathbf{L}_0$ represents the variation of the design velocity gradient in the undeformed configuration $\mathcal{B}_0$ and is given by

$$\delta\mathbf{L}_0 = \delta\mathbf{F}_R \cdot \mathbf{F}_R^{-1} = \frac{\partial \delta\mathbf{x}_0}{\partial \mathbf{x}_0} \tag{38}$$

where $\mathbf{F}_R = \frac{\partial \mathbf{x}_0}{\partial \boldsymbol{\eta}}$ is the mapping of material points from a reference domain $\mathcal{B}_R$ to the undeformed configuration $\mathcal{B}_0$. This reference domain shown in fig. 2 is assumed to be independent of both the design parameters and the deformation history of the workpiece (a unit square domain, for example) and is typically used when dealing with preform shape design problems [2]. In eq. (38), $\mathbf{x}_0$ defines the initial geometry of the preform (see fig. 1) whose shape is specified by the design parameters $\boldsymbol{\beta}_p$. Hence, $\delta\mathbf{L}_0$ is a function of these design variables. If the reference domain $\mathcal{B}_R$ coincides with the undeformed configuration $\mathcal{B}_0$, then $\mathbf{F}_R = \mathbf{I}$. This gives $\delta\mathbf{L}_0 = \mathbf{0}$. With this value, the sensitivity boundary value problem, eqs.(35)-(37), reduces to

$$\begin{aligned}
\nabla_0 \cdot \delta\mathbf{P} + \delta\rho_0\mathbf{b} &= 0 && \text{in } V_0 && (39)\\
\delta\mathbf{P} \cdot \mathbf{n}_0 &= \delta\bar{\mathbf{t}}_0 + (\delta\mathbf{F}_u \cdot \mathbf{F}_u^{-1}) : (\mathbf{I} - \mathbf{n}\otimes\mathbf{n})\bar{\mathbf{t}}_0 && \text{on } A_0^\sigma && (40)\\
\delta\mathbf{u} &= \delta\bar{\mathbf{u}} && \text{on } A_0^u && (41)
\end{aligned}$$

Equations (39)-(40) are mainly used when dealing with die design problems while keeping the preform shape fixed [3].

Using the definition of $\mathbf{P} = det(\mathbf{F}_u)\boldsymbol{\sigma} \cdot \mathbf{F}_u^{-T}$ and the relations $\boldsymbol{\sigma} = \mathbf{R} \cdot \bar{\boldsymbol{\sigma}} \cdot \mathbf{R}^T$, $\mathbf{F} = \mathbf{R} \cdot \mathbf{U}$, $\mathbf{F}_u = \mathbf{F} \cdot \mathbf{F}_N$, one can show that [3]

$$\begin{aligned}
\delta\mathbf{P} &= det(\mathbf{F}_u)[tr(\delta\mathbf{F}_u \cdot \mathbf{F}_u^{-1})\boldsymbol{\sigma} + \delta\boldsymbol{\sigma} - \boldsymbol{\sigma} \cdot (\delta\mathbf{F}_u \cdot \mathbf{F}_u^{-1})^T] \cdot \mathbf{F}_u^{-T} && (42)\\
\delta\boldsymbol{\sigma} &= \mathbf{R} \cdot \delta\bar{\boldsymbol{\sigma}} \cdot \mathbf{R}^T + \delta\mathbf{R} \cdot \mathbf{R}^T \cdot \boldsymbol{\sigma} - \boldsymbol{\sigma} \cdot \delta\mathbf{R} \cdot \mathbf{R}^T && (43)\\
\delta\mathbf{R} \cdot \mathbf{R}^T &= \delta\mathbf{F} \cdot \mathbf{F}^{-1} - \mathbf{R} \cdot \delta\mathbf{U} \cdot \mathbf{F}^{-1} && (44)\\
\delta\mathbf{U} &= [\mathbf{U}^{-1} \cdot (\mathbf{F}^T \cdot \delta\mathbf{F})_s]_s && (45)\\
\delta\mathbf{F} &= \delta\mathbf{F}_u \cdot \mathbf{F}_u^{-1} - \mathbf{F}_u \cdot \mathbf{F}_N^{-1} \cdot \delta\mathbf{F}_N \cdot \mathbf{F}_N^{-1} && (46)
\end{aligned}$$

where $\delta\mathbf{F}_u$ is related to the displacement variation $\delta\mathbf{u}$ by

$$\delta\mathbf{F}_u = \frac{\partial \delta\mathbf{u}}{\partial \mathbf{x}_0} - (\mathbf{F}_u - \mathbf{I}) \cdot \delta\mathbf{L}_0 \tag{47}$$

This formulation is complemented with the sensitivity equations for the constitutive model which relates the stress variation $\delta\bar{\boldsymbol{\sigma}}$ to the strain variation $\delta\Delta\bar{\mathbf{E}}$. In this work, these equations are derived from the integrated form of the constitutive equations (20)-(23). Omitting details of the derivation, one obtains

$$\begin{aligned}
\delta\bar{\boldsymbol{\sigma}} &= \bar{\boldsymbol{\mathcal{L}}} : \delta\Delta\bar{\mathbf{E}} + \bar{\boldsymbol{\mathcal{A}}} : \delta\boldsymbol{\sigma}_N + \delta\alpha\bar{\mathbf{N}} && (48)\\
\delta s &= \frac{c}{b_2}(b_1\delta\bar{\sigma}* + a_1\delta s_N) && (49)
\end{aligned}$$

where $\delta\boldsymbol{\sigma}_N$ and δs_N represent the variation of the Cauchy stress and the hardness at time t_N, respectively, and

$$\begin{aligned}
\bar{\boldsymbol{\mathcal{A}}} &= \eta\boldsymbol{\mathcal{I}} + \frac{1-\eta}{3}\mathbf{I}\otimes\mathbf{I} - (\eta - c)\bar{\mathbf{N}}\otimes\bar{\mathbf{N}} && (50)\\
\delta\alpha &= -\sqrt{\frac{2}{3}}\frac{c\,a_2}{b_2}\delta s_N && (51)\\
\delta\Delta\bar{\mathbf{E}} &= 4(\mathbf{U}+\mathbf{I})^{-1} \cdot \delta\mathbf{U} \cdot (\mathbf{U}+\mathbf{I})^{-1} && (52)
\end{aligned}$$

As mentioned before, the sensitivity problem, Eqs.(35)-(52), is linear in $\delta\mathbf{u}$ and, hence, its solution does not require an iterative procedure. Also, note that the solution of the sensitivity equations is carried out in parallel with the solution of the direct problem.

When solving preform and die design problems, the design parameters $\boldsymbol{\beta}_p$ and $\boldsymbol{\beta}_d$ are introduced in the solution procedure of either eqs.(35)-(36) or eqs.(39)-(40) in two ways:

1. In the variation of the design velocity gradient $\delta\mathbf{L}_0$ which is a function of the preform geometry, and
2. In the traction boundary conditions along the workpiece-die interface which depends on the die geometry.

For the first case, we note that the mapping $\mathbf{F}_R$, defining the reference domain $\mathcal{B}_R$, accounts for the geometric representation of the preform, and hence, is expressed in terms of the design parameters $\boldsymbol{\beta}_p$. As an example of this mapping, consider the following transfinite mapping [11] of the preform to a unit square (see fig.1),

$$x_{01} = \eta_1 G(\eta_2) \tag{53}$$
$$x_{02} = \eta_2 d \tag{54}$$

where $G(\eta_2) = \Sigma\beta_{pi} g_i(\eta_2)$, with $g_i(\eta_2)$ as basis functions, defines the parametric representation of the free boundary of the workpiece, β_{pi} is the i-th design parameter, and $2d$ is the height of the preform. From this mapping, we can derive

$$\mathbf{F}_R = \frac{\partial \mathbf{x}_0}{\partial \boldsymbol{\eta}} = \begin{bmatrix} G & \eta_1 \frac{dG}{d\eta_2} \\ 0 & d \end{bmatrix}$$

and hence

$$\delta\mathbf{F}_R = \frac{\partial \mathbf{F}_R}{\partial \beta_{pk}} \delta\beta_{pk} = \begin{bmatrix} g_k & \eta_1 \frac{dg_k}{d\eta_2} \\ 0 & 0 \end{bmatrix} \delta\beta_{pk}$$

Therefore, the expression for $\delta\mathbf{L}_0 = \delta\mathbf{F}_R \cdot \mathbf{F}_R^{-1}$ is

$$\delta\mathbf{L}_0 = \begin{bmatrix} g_k & \frac{\eta_1}{d}(G\frac{dg_k}{d\eta_2} - \frac{dG}{d\eta_2} g_k) \\ 0 & 0 \end{bmatrix} \frac{\delta\beta_{pk}}{G}$$

The sensitivity equations (35)-(37) will then have to be solved for each β_{pk} to obtain the sensitivity of the field variables to $\boldsymbol{\beta}_p$.

For the second case, consider the traction boundary condition along the interface workpiece-die. Assuming a Coulomb friction model, we can write for the applied traction $\bar{\mathbf{t}}$,

$$\bar{\mathbf{t}} = -p\,\mathbf{n} \pm \mu_f\, p\,\mathbf{s} \tag{55}$$

where p in the normal traction (pressure) to the contact surface, μ_f is the friction coefficient, and $\mathbf{n}$ and $\mathbf{s}$ are unit vectors normal to and along the interface respectively. The specific sign in front of μ_f is chosen according to the direction of motion of the workpiece with respect the die. Taking the design variation of eq.(55) with respect to the design parameters $\boldsymbol{\beta}_d$, and using the relations: $\mathbf{n}\cdot\mathbf{n} = 1$, $\mathbf{n}\cdot\mathbf{s} = 0$, one obtains

$$\delta\bar{\mathbf{t}} = -[\delta p \pm \mu_f\, p(\delta\mathbf{n}\cdot\mathbf{s})]\mathbf{n} + [-p(\delta\mathbf{n}\cdot\mathbf{s}) \pm \mu_f \delta p]\mathbf{s} \tag{56}$$

Using a parametric representation for the die surface of the form $x_2 = h(x_1, \boldsymbol{\beta}_d)$, one can show that [2]

$$\delta\mathbf{n}\cdot\mathbf{s} = \frac{h_{,1}\delta h_{,2} - h_{,2}\delta h_{,1}}{h_{,1}^2 + h_{,2}^2} \tag{57}$$

where $h_{,i} = \frac{\partial h}{\partial x_i}$ and $\delta h_{,i}$ is its corresponding design variation. Note that the changes in the die surface alters only the unit normal $\mathbf{n}$. By substitution of eq.(57) into eqs.(39)-(41), we obtain an expression for the design variation of the traction $\delta\bar{\mathbf{t}}$ along the interface as a function of the design parameters $\boldsymbol{\beta}_d$. In this case, then, eqs.(39)-(41) need to be solved for each parameter $\boldsymbol{\beta}_d$ to determine the sensitivity of the field variables to the die shape.

3.4 Methodology for Optimization-Based Design of Multi-Step Forging Processes

The ideas behind the methodology will be developed by considering the shape design optimization of the two-step forging process represented schematically in fig. 1. Here the purpose is to compute the field variables $(\mathbf{u}, \mathbf{P})$ and their sensitivities $(\delta\mathbf{u}, \delta\mathbf{P})$ with respect to the design parameters $(\boldsymbol{\beta}_p, \boldsymbol{\beta}_d)$ at the end of the two-step process. This information is then used by the optimizer to find an optimal design of the preform and die geometries.

This problem is solved using eqs.(10)-(12) for the direct problem together with eqs.(35)-(37) and eqs.(39)-(41) for the sensitivity analysis. Computing the sensitivity field $(\delta\mathbf{u}, \delta\mathbf{P})$ with respect to $\boldsymbol{\beta}_p$ during the two-step forging can be considered as a preform design process. Similarly, obtaining the sensitivities $(\delta\mathbf{u}, \delta\mathbf{P})$ with respect to $\boldsymbol{\beta}_d$ (second forging step) can be seen as a die design problem. Note also that, during the second forging step, the total deformation gradient $\mathbf{F}_u$ in eqs.(35)-(37) will be the composition $\mathbf{F}_u^{(2)} \cdot \mathbf{F}_u^{(1)}$, and hence the deformation gradient $\mathbf{F}_u^{(1)}$ needs to be saved from the first forging step. In essence, the methodology to solve this problem is as follows:

1. Guess the initial values of the preform design parameters $\boldsymbol{\beta}_p$.
2. Do i = 1, # steps1 (first forging step)
   (a) Solve the direct problem for $(\mathbf{u}, \mathbf{P})$. Use eqs.(10)-(12).
   (b) Solve n_p sensitivity problems for the variations $(\delta\mathbf{u}, \delta\mathbf{P})_{\beta_p}$, where n_p is the number of preform design parameters. Use eqs.(35)-(37).
3. At end of first forging step, the sensitivities $(\delta\mathbf{u}, \delta\mathbf{P})_{\beta_p}$ will be used as initial conditions for solving the second forging step.
4. Guess the initial values of the die design parameters $\boldsymbol{\beta}_d i$, or use previous computed value.
5. Do i = 1, # steps2 (second forging step)
   (a) Solve the direct problem for $(\mathbf{u}, \mathbf{P})$. Use eqs.(10)-(12).
   (b) Solve n_p sensitivity problems for the variations $(\delta\mathbf{u}, \delta\mathbf{P})_{\beta_p}$. Use eqs.(35)-(37).
   (c) Solve n_d sensitivity problems for the variations $(\delta\mathbf{u}, \delta\mathbf{P})_{\beta_d}$, where n_d is the number of die design parameters. Use eqs.(39)-(41).
6. Use an optimization algorithm to compute new values of $(\boldsymbol{\beta}_p, \boldsymbol{\beta}_d)$. Here, the objective function Φ is minimized using gradient information $(\delta_{\beta_p}\Phi, \delta_{\beta_d}\Phi)$.
7. Check convergence of the objective function:
   If $\| \Phi \| \leq$ Tolerance $\Rightarrow$ exit
   If $\| \Phi \| >$ Tolerance $\Rightarrow$ go to step 2 with the new values of $\boldsymbol{\beta}_p$ and $\boldsymbol{\beta}_d$.

4 A Computational Tool for Solving Optimization-Based Design Problems in Mechanics: PDESolve

4.1 General Features of PDESolve

PDESolve is an object-oriented class library constructed using the C++ language and designed for solving complex partial differential equations (PDE) with general tensor value functions. It forms the basis for an open-standard computer language for scientific, engineering, and financial PDE applications. When using PDESolve the user specifies the problem to be solved in a high-level form, together with the discretization methods and the solution procedures. This mathematical level of abstraction gives the user a fully flexible programming language for constructing PDE simulations rapidly. Besides, the object-oriented architecture of PDESolve allows a user to incorporate new capabilities into an existing PDESolve code in a clean and efficient manner.

The basic C++ classes in PDESolve are:

- `DiffOp`: represents differential operators
- `Expr`: builds mathematical expressions including PDEs
- `SpatialDomain`: specifies the problem domain (geometry)
- `BC`: prescribes boundary conditions

- `FDMesh`: specifies the finite difference mesh (spatial discretization)
- `DiscExpr`: constructs the discrete form of the PDE using the mesh and the boundary conditions
- `Function`: represents both known and unknown functions

Differential Operators and Expressions: In PDESOLVE, the partial differential operator $\partial^2/\partial x^2$ is represented as:

```
DiffOp dxx(2,0,4);
```

In the argument list, the first integer specifies the order of the differential operator, the second integer, the index of the independent variable (i.e. x is taken to be independent variable # 0), and the third integer, the order of accuracy desired in the numerical representation of the differential operator. Currently, up to fourth-order accuracy is supported for the differential operators. Optionally, a user defined stencil family file may be provided. The syntax for defining the 3-D Laplace's equation to third-order accuracy is

```
DiffOp dxx(2,0,3);
DiffOp dyy(2,1,3);
DiffOp dzz(2,2,3);
Expr laplace = (dxx + dyy + dzz) * u;
```

Geometry Specification: PDESOLVE uses a *multi-block body-fitted coordinate representation* of the problem geometry. This representation is based on two binary operators, "*" (Cartesian product) and "+" (union), together with a base operator for creating line segments `SpatialDomain(<high>,<low>)`. With this representation, the geometry and discretization are specified unambiguously. The PDESOLVE syntax for defining a 1D domain (0,1) is

```
SpatialDomain geom(0.0,1.0);
```

A unit cube is specified as

```
SpatialDomain geom = SpatialDomain(0.0,1.0)
                   * SpatialDomain(0.0,1.0)
                   * SpatialDomain(0.0,1.0);
```

More complicated shapes can be generated by adding together components such as in the following definition of an "L" shaped geometric domain:

```
SpatialDomain geom = SpatialDomain(0.0,1.0) * SpatialDomain(0.0,1.0) +
                     SpatialDomain(1.0,2.0) * SpatialDomain(0.0,1.0) +
                     SpatialDomain(0.0,1.0) * SpatialDomain(1.0,2.0);
```

PDESOLVE can be used to solve problems on body-fitted meshes around complex shapes as well. This is done through a user-specified transformation map that could be analytical or numerical.

Boundary Conditions: For the line segment, to specify $u = 0$ at the left boundary and to leave the right-boundary value unconstrained, the syntax is:

```
BC boundaryCond(u==0, Unc());
```

For the unit cube, boundary conditions can be of form

```
BC bc = BC(u==0, u==1)
      * BC(u==0, Dy*u==0)
      * BC(u==0, Dz*u==0);
```

where the `DiffOp`'s Dy and Dz are used to specify Neumann boundary conditions. The PDESOLVE boundary condition language is based on structural parallelism with the geometry specification. This parallel structure provides a well-defined specification of the boundary conditions.

DiscExpr: Discretized Version of PDE Problem: Once the PDE, mesh and boundary conditions have been specified, the discretized problem is obtained using the statement

```
DiscExpr discprob(laplace, grid, u, bcs);
```

The above command causes the discrete equations to be built and stored as a `DiscExpr` object called `discprob`. This system can be solved for u at the mesh points simply by specifying

```
discprob.solve();
```

Simple Example in PDESOLVE**:** We now consider a simple nonlinear PDE problem to illustrate how the above C++ classes are put together to solve a PDE problem. We solve

$$\frac{\partial^2 u}{\partial x^2} + u\frac{\partial u}{\partial x} = f(x), \tag{58}$$

on the interval $[0, 1]$, where $f(x) = x(x - 10) - 4$, with the boundary conditions being $u|_0 = 0, u|_1 = 1$. The executable PDESOLVE code for this problem is:

```
#include "pdesolve.h"
Real fFunc(const Coords &X) {
  return X[0]*(X[0] - 10.0) -4};
void main() {
  SpatialDomain domain(0, 1);
  FDMesh grid(domain,FDGrid(11));
  Function u;
  Function f(1,scalar,fFunc);
  BC bc(u==0, u==1);
  DiffOp dx(1,0,2);
  DiffOp dxx(2,0,2);
  Expr eq = dxx*u + u*dx*u - f;
  DiscExpr discExpr(eq, grid, bc, NewtonSolver(40,1e-5));
  u = discExpr.solve();  //Solve for u
  cout << u << endl;
  }
```

Note that here we specified that the non-linear problem be solved with Newton's method. The linearization is performed automatically (although the user can input the linearization and loop "by hand") with the maximum number of iterations being 40 and the tolerance being 10^{-5}. Any other nonlinear solver can be used by suitably specifying it in the last argument of the `DiscExpr`. This argument is optional (with Newton linearization being the default) and is not present for linear problems.

If we instead need to solve the PDE

$$\frac{\partial^3 u}{\partial x^3} + u\frac{\partial u}{\partial x} = f(x), \tag{59}$$

the only line that needs to be changed is

```
Expr eq = dxxx*u + u*dx*u;
```

remembering to include the definition for `dxxx` and to provide an additional boundary condition:

```
BC bc(u==0 && dx*u==1, u==1);.
```

This will specify that on the left side of the domain `u=0` and `dx*u=1`.

In the next sections, we use PDESOLVE to solve optimization-based design problems in linear elasticity and small deformation plasticity. We will apply the general methodology developed for optimization-based design problems to both cases and show how PDESOLVE can be used as an effective numerical tool to solve this kind of problems. As mentioned before, we will use the differential (strong) form of the governing equations which are amenable to the finite difference method (current version of PDESOLVE).

4.2 Design Optimization in Linear Elasticity

4.2.1 Problem Formulation

For linear elastic problems, the different configurations of the body shown in fig. 2 will coincide, and hence, $\mathcal{B}_0 = \mathcal{B}_N = \mathcal{B}_{N+1}$ ($\mathbf{x}_0 = \mathbf{x}_N = \mathbf{x}_{N+1} = \mathbf{x}$). For this case, the strain and stress measures are given by the small elastic strain $\boldsymbol{\epsilon}$ and the Cauchy stress $\boldsymbol{\sigma}$. Hence, the general performance functional, Eq.(4), can be written as

$$\Phi(\boldsymbol{\beta}) = \int_V f(\mathbf{u}(\boldsymbol{\beta}), \boldsymbol{\epsilon}(\boldsymbol{\beta}), \boldsymbol{\sigma}(\boldsymbol{\beta}))dV \tag{60}$$

As before, for optimizing this functional we need to solve both the direct problem for the quantities $(\mathbf{u}, \boldsymbol{\epsilon}, \boldsymbol{\sigma})$ and the sensitivity problem for the variations $(\delta\mathbf{u}, \delta\boldsymbol{\epsilon}, \delta\boldsymbol{\sigma})$.

For an isothermal, quasi-static deformation process, the direct problem of linear elasticity is governed by the equilibrium equation and boundary conditions written as

$$\begin{aligned} \boldsymbol{\nabla}\cdot\boldsymbol{\sigma} + \mathbf{f} &= \mathbf{0} \quad \text{in } V & (61)\\ \boldsymbol{\sigma}\cdot\mathbf{n} &= \bar{\mathbf{t}} \quad \text{on } A_\sigma & (62)\\ \mathbf{u} &= \bar{\mathbf{u}} \quad \text{on } A_u & (63) \end{aligned}$$

together with the constitutive material response given by

$$\boldsymbol{\sigma} = \boldsymbol{\mathcal{C}}:\boldsymbol{\epsilon} \quad , \quad \boldsymbol{\epsilon} = \frac{1}{2}(\boldsymbol{\nabla}\mathbf{u} + (\boldsymbol{\nabla}\mathbf{u})^T) \tag{64}$$

In these equations, $\mathbf{f}$ is the body force per unit volume, $\mathbf{n}$ is a unit normal, $\bar{\mathbf{t}}$ and $\bar{\mathbf{u}}$ are prescribed tractions and displacements on the surfaces A_σ and A_u, respectively, and $\boldsymbol{\mathcal{C}}$ is the constant elasticity matrix. Note that these equations are linear and, hence, their solution can be obtained in one step.

The sensitivity problem is formulated using the direct differentiation method. Here the sensitivity equations are obtained by taking the variations of the equations governing the direct problem, eqs.(61)-(64), to give, for the equilibrium equations and boundary conditions

$$\begin{aligned} \boldsymbol{\nabla}\cdot\delta\boldsymbol{\sigma} - \boldsymbol{\nabla}\boldsymbol{\sigma}:\delta\mathbf{L}^{\mathrm{T}} + \delta\mathbf{f} &= \mathbf{0} \quad \text{in } V & (65)\\ (\delta\boldsymbol{\sigma} - \boldsymbol{\sigma}\cdot\delta\mathbf{L}^{\mathrm{T}})\cdot\mathbf{n} + (\delta\mathbf{L}:\mathbf{n}\otimes\mathbf{n})\bar{\mathbf{t}} &= \delta\bar{\mathbf{t}} \quad \text{on } A_\sigma & (66)\\ \delta\mathbf{u} &= \delta\bar{\mathbf{u}} \quad \text{on } A_u & (67) \end{aligned}$$

and for the elastic constitutive response

$$\delta\boldsymbol{\sigma} = \boldsymbol{\mathcal{C}}:\delta\boldsymbol{\epsilon} \quad , \quad \delta\boldsymbol{\epsilon} = \frac{1}{2}[\boldsymbol{\nabla}\delta\mathbf{u} + (\boldsymbol{\nabla}\delta\mathbf{u})^T - (\boldsymbol{\nabla}\mathbf{u}\cdot\delta\mathbf{L} + (\boldsymbol{\nabla}\mathbf{u}\cdot\delta\mathbf{L})^T)] \tag{68}$$

where $\delta\mathbf{L} = \delta\mathbf{F}_R\cdot\mathbf{F}_R^{-1} = \frac{\partial\delta\mathbf{x}}{\partial\mathbf{x}}$. In this case, the design parameters $\boldsymbol{\beta}$ are introduced in the problem through the mapping $\mathbf{F}_R$ (we use transfinite transformations for this purpose).

4.2.2 Implementation of 2-D Linear Elasticity Problems in PDESolve

The foregoing equations, eqs.(61)-(68), have been simplified for the case of two-dimensional (plane stress and plane strain) problems. The expanded form of these equations is given by:

Direct Problem

Equilibrium equations,

$$\begin{aligned} \sigma_{11,1} + \sigma_{12,2} + f_1 &= 0\\ \sigma_{21,1} + \sigma_{22,2} + f_2 &= 0 \end{aligned}$$

Boundary conditions,

$$\begin{aligned} \sigma_{11}n_1 + \sigma_{12}n_2 &= \bar{t}_1\\ \sigma_{21}n_1 + \sigma_{22}n_2 &= \bar{t}_2\\ u_1 &= \bar{u}_1\\ u_2 &= \bar{u}_2 \end{aligned}$$

Constitutive law,

$$\begin{aligned}\sigma_{11} &= (2\mu+\bar{\lambda})\epsilon_{11}+\bar{\lambda}\epsilon_{22}\\ \sigma_{22} &= \bar{\lambda}\epsilon_{11}+(2\mu+\bar{\lambda})\epsilon_{22}\\ \sigma_{12} &= 2\mu\epsilon_{12}\end{aligned}$$

where $\bar{\lambda}=\lambda$ for plane strain, and $\bar{\lambda}=\frac{2\mu\lambda}{2\mu+\lambda}$ for plane stress,
Kinematics relations,

$$\begin{aligned}\epsilon_{11} &= u_{1,1}\\ \epsilon_{22} &= u_{2,2}\\ \epsilon_{12} &= \frac{1}{2}(u_{1,2}+u_{2,1})\end{aligned}$$

Sensitivity Problem

Equilibrium equations,

$$\begin{aligned}\delta\sigma_{11,1}+\delta\sigma_{12,2}-\sigma_{11,1}\delta L_{11}-\sigma_{11,2}\delta L_{21}-\sigma_{12,1}\delta L_{12}-\sigma_{12,2}\delta L_{22}+\delta f_1 &= 0\\ \delta\sigma_{21,1}+\delta\sigma_{22,2}-\sigma_{21,1}\delta L_{11}-\sigma_{21,2}\delta L_{21}-\sigma_{22,1}\delta L_{12}-\sigma_{22,2}\delta L_{22}+\delta f_2 &= 0\end{aligned}$$

Boundary conditions,

$$\begin{aligned}\delta\sigma_{11}n_1+\delta\sigma_{12}n_2+(\sigma_{12}n_1-\sigma_{11}n_2)[(\delta L_{11}n_2-\delta L_{12}n_1)n_1+(\delta L_{21}n_2-\delta L_{22}n_1)n_2] &= \delta\bar{t}_1\\ \delta\sigma_{21}n_1+\delta\sigma_{22}n_2+(\sigma_{22}n_1-\sigma_{21}n_2)[(\delta L_{11}n_2-\delta L_{12}n_1)n_1+(\delta L_{21}n_2-\delta L_{22}n_1)n_2] &= \delta\bar{t}_2\\ \delta u_1=\delta\bar{u}_1\\ \delta u_2=\delta\bar{u}_2\end{aligned}$$

Constitutive law,

$$\begin{aligned}\delta\sigma_{11} &= (2\mu+\bar{\lambda})\delta\epsilon_{11}+\bar{\lambda}\delta\epsilon_{22}\\ \delta\sigma_{22} &= \bar{\lambda}\delta\epsilon_{11}+(2\mu+\bar{\lambda})\delta\epsilon_{22}\\ \delta\sigma_{12} &= 2\mu\delta\epsilon_{12}\end{aligned}$$

Kinematics relations,

$$\begin{aligned}\delta\epsilon_{11} &= \delta u_{1,1}-u_{1,1}\delta L_{11}-u_{1,2}\delta L_{21}\\ \delta\epsilon_{22} &= \delta u_{2,2}-u_{2,1}\delta L_{12}-u_{2,2}\delta L_{22}\\ \delta\epsilon_{12} &= \frac{1}{2}(\delta u_{1,2}+\delta u_{2,1})-\frac{1}{2}(u_{1,1}\delta L_{12}+u_{1,2}\delta L_{22}+u_{2,1}\delta L_{11}+u_{2,2}\delta L_{21})\end{aligned}$$

The implementation of these equations has been carried out in PDESolve using existing C++ classes. The corresponding modules are shown below. The specified boundary conditions presented in this program correspond to the case of a thin circular plate with a hole (plane stress) loaded along the outer radius (see Fig. 3). For this problem, $\delta L_{ij}=\frac{\partial\delta x_i}{\partial x_j}$ is computed using the transfinite bilinear mapping,

$$\begin{aligned}x_1 &= [a+\eta_2(b-a)]\cos(\frac{\eta_1\pi}{2})\\ x_2 &= [a+\eta_2(b-a)]\sin(\frac{\eta_1\pi}{2})\end{aligned}$$

where $a(=\beta_1)$ and $b(=\beta_2)$ are the inner radius and outer radius of the plate, respectively. This program has been used to predict the sensitivity of the radial and tangential stresses, $\delta\sigma_{rr}$ and $\delta\sigma_{\theta\theta}$, to variations of the inner radius (β_1). Comparison of computed results with the analytic solution is shown in Fig. 4.

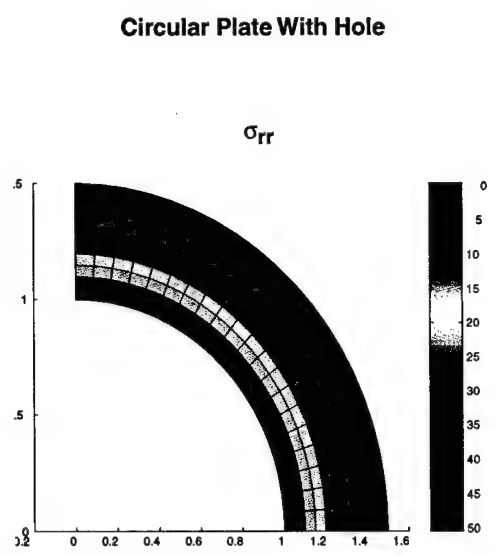

Figure 3: Circular plate with a hole.

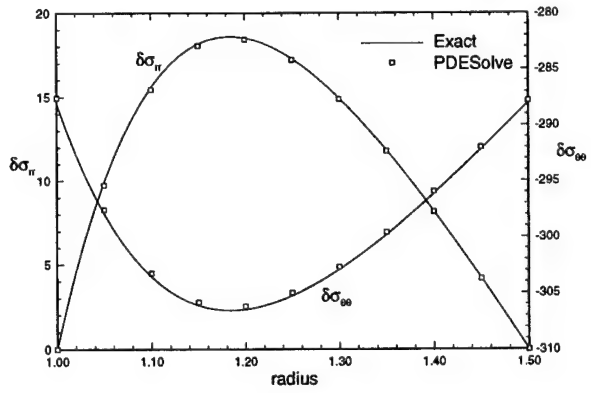

Figure 4: Stress sensitivities for circular plate with a hole.

```
void Elasticity_Direct()
{ \* U : displacement vector, (U0,U1)
     E : elastic strain tensor, (E00,E11,E01)
     T : Cauchy stress tensor, (T00,T11,T01)
     F : body force vector, (F0,F1)
     Po : load applied to outer radius, (Po0,Po1)
     M : 3x3 elasticity matrix (plane stress) *\

  SpatialDomain computDomain=SpatialDomain(0.,1.)*SpatialDomain(0.,1.);
  Function fwdMap(2,Tensor(2,1),fwdMapFunc),Function(2,Tensor(2,2),DfwdMapFunc));
  Function invMap(2,Tensor(2,1),invMapFunc);
  SpatialDomain physDomain=SpatialDomain(computDomain,Map(fwdMap,invMap));
  FDMesh grid=FDMesh(physDomain,FDMeshSpec(nX)*FDMeshSpec(nY));

  DiffOp Dx(1,0,4);
  DiffOp Dy(1,1,4);

  Function U(2,Tensor(2,1));
  Function E(2,Tensor(3,1));
  Function T(2,Tensor(3,1));
  Function F(2,Tensor(2,1),CFuncF);
  Function Po(2,Tensor(2,1),CFuncPo);
  Function M(2,Tensor(2,3),CFuncM);

  Expr diffOperDirect=List(List(Dx,0.0,Dy),List(0.0,Dy,Dx));
  Expr E=transpose(diffOperDirect)*U;
  Expr T=M*E;
  Expr equilEqns=diffOperDirect*T+F;
  BC bc=BC(T[0]*Normal(0)+T[2]*Normal(1)==0.0 && U[1]==0.0,
           U[0]==0.0 && T[2]*Normal(0)+T[1]*Normal(1)==0.0)
      * BC(T[0]*Normal(0)+T[2]*Normal(1)==0.0 &&
           T[2]*Normal(0)+T[1]*Normal(1)==0.0,
           T[0]*Normal(0)+T[2]*Normal(1)==Po[0] &&
           T[2]*Normal(0)+T[1]*Normal(1)==Po[1]);
  DiscExpr elastDirect(equilEqns, grid, U, bc);
  U = elastDirect.solve();
}
```

```
void Elasticity_Sensitivity()
{ \*
     vU : variation of displacement, (vU0,vU1)
     vE : variation of strain, (vE00,vE11,vE01)
     vT : variation of stress, (vT00,vT11,vT01)
     vF : variation of body force, (vF0,vF1)
     vPo : variation of applied traction, (vPo0,vPo1)
     vL : variation of the design velocity gradient, ((vL00,vL01),(vL10,vL11))
     M : 3x3 elasticity matrix (plane stress)
  *\
  SpatialDomain computDomain=SpatialDomain(0,1)*SpatialDomain(0,1);
  Function fwdMap(2,Tensor(2,1),fwdMapFunc,Function(2,Tensor(2,2),DfwdMapFunc));
  Function invMap(2,Tensor(2,1),invMapFunc);
  SpatialDomain physDomain(computDomain, Map(fwdMap,invMap));
  FDMesh grid=FDMesh(physDomain,FDMeshSpec(nX)*FDMeshSpec(nY));

  DiffOp Dx(1,0,4);
  DiffOp Dy(1,1,4);

  Function vU(2,Tensor(2,1));
  Function vE(2,Tensor(3,1));
  Function vT(2,Tensor(3,1));
  Function vF(2,Tensor(2,1),CVFuncF);
  Function vPo(2,Tensor(2,1),CVFuncPo);
  Function vL(2,Tensor(2,2),CVFuncL);
  Function M(2,Tensor(2,3),CFuncM);

  Expr operL1=List(List(vL[0][0],0.0,vL[0][1]),List(0.0,vL[0][1],vL[0][0]));
  Expr operL2=List(List(vL[1][0],0.0,vL[1][1]),List(0.0,vL[1][1],vL[1][0]));
  Expr diffOperDirect=List(List(Dx,0.0,Dy),List(0.0,Dy,Dx));
  Expr diffOperSensit=operL1*Dx+operL2*Dy;
  Expr vE=transpose(diffOperDirect)*vU-transpose(diffOperSensit)*U;
  Expr vT=M*vE;
  Expr vEquilEqns=diffOperDirect*vT-diffOperSensit*T+vF;
  BC vBC=BC(vT[0]*Normal(0)+vT[2]*Normal(1)+(T[2]*Normal(0)-T[0]*Normal(1))*
              ((vL[0][0]*Normal(1)-vL[0][1]*Normal(0))*Normal(0)+
               (vL[1][0]*Normal(1)-vL[1][1]*Normal(0))*Normal(1))==0.0 &&
             vU[1]==0.0,
             vU[0]==0.0 &&
             vT[2]*Normal(0)+vT[1]*Normal(1)+(T[1]*Normal(0)-T[2]*Normal(1))*
              ((vL[0][0]*Normal(1)-vL[0][1]*Normal(0))*Normal(0)+
               (vL[1][0]*Normal(1)-vL[1][1]*Normal(0))*Normal(1))==0.0))
        * BC(vT[0]*Normal(0)+vT[2]*Normal(1)+(T[2]*Normal(0)-T[0]*Normal(1))*
              ((vL[0][0]*Normal(1)-vL[0][1]*Normal(0))*Normal(0)+
               (vL[1][0]*Normal(1)-vL[1][1]*Normal(0))*Normal(1))==0.0 &&
             vT[2]*Normal(0)+vT[1]*Normal(1)+(T[1]*Normal(0)-T[2]*Normal(1))*
              ((vL[0][0]*Normal(1)-vL[0][1]*Normal(0))*Normal(0)+
               (vL[1][0]*Normal(1)-vL[1][1]*Normal(0))*Normal(1))==0.0,
             vT[0]*Normal(0)+vT[2]*Normal(1)+(T[2]*Normal(0)-T[0]*Normal(1))*
              ((vL[0][0]*Normal(1)-vL[0][1]*Normal(0))*Normal(0)+
               (vL[1][0]*Normal(1)-vL[1][1]*Normal(0))*Normal(1))==vPo[0] &&
             vT[2]*Normal(0)+vT[1]*Normal(1)+(T[1]*Normal(0)-T[2]*Normal(1))*
              ((vL[0][0]*Normal(1)-vL[0][1]*Normal(0))*Normal(0)+
               (vL[1][0]*Normal(1)-vL[1][1]*Normal(0))*Normal(1))==vPo[1]));
  DiscExpr elastSensit(vEquilEqns,grid,vU,vBC);
  vU=elastSensit.solve();
}
```

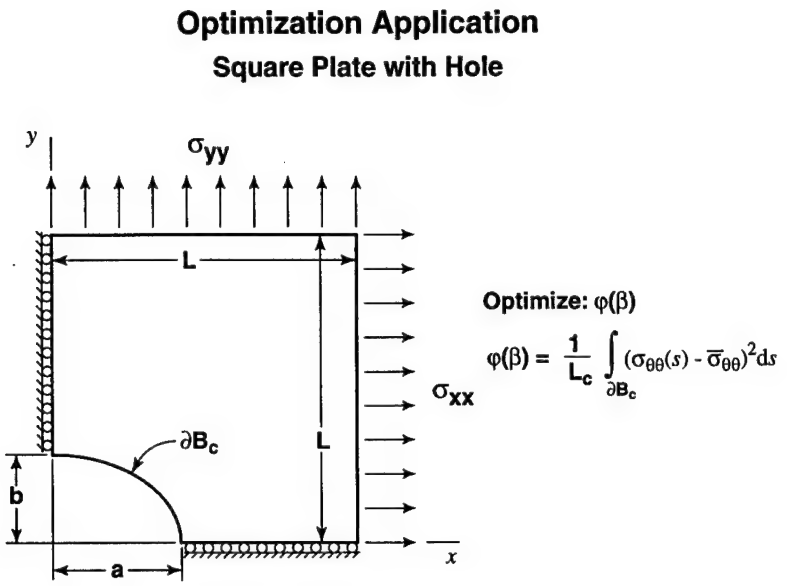

Figure 5: Schematic representation of optimization problem.

4.2.3 Application of PDESOLVE to Optimization Problems in Elasticity

As an application of optimization-based design in linear elasticity, in this section we compute the optimal shape of the cutout in a thin elastic plate subjected to constant ratios of far field stresses (see Fig. 5). Specifically, we want to determine the optimal shape of the cutout such that, for the given far field stresses, we obtain a uniform distribution of tangential stresses along the boundary of the cutout.

The particular objective function to be minimized in this problem is given by

$$\phi = \frac{1}{L_c}\int_{\partial B_c}(\sigma_{\theta\theta} - \bar{\sigma}_{\theta\theta})^2 dl$$

where $\sigma_{\theta\theta}$ is the tangential stress along the cutout, and

$$\bar{\sigma}_{\theta\theta} = \frac{1}{L_c}\int_{\partial B_c}\sigma_{\theta\theta}\, dl \quad , \quad L_c = \int_{\partial B_c} dl$$

The variation of ϕ with respect to the design variable β_i ($\delta\phi = \frac{\partial\phi}{\partial\beta_i}\delta\beta_i$) is expressed as

$$\delta\phi = \frac{2}{L_c}\int_{\partial B_c}(\sigma_{\theta\theta} - \bar{\sigma}_{\theta\theta})(\delta\sigma_{\theta\theta} - \delta\bar{\sigma}_{\theta\theta})dl + \frac{1}{L_c}\int_{\partial B_b}(\sigma_{\theta\theta} - \bar{\sigma}_{\theta\theta})^2\delta dl - \frac{\phi}{L_c}\delta L_c$$

where

$$\delta\bar{\sigma}_{\theta\theta} = \frac{1}{L_c}\int_{\partial B_c}(\delta\sigma_{\theta\theta}dl + \sigma_{\theta\theta}\delta dl) - \frac{\bar{\sigma}_{\theta\theta}}{L_c}\delta L_c$$

$$\delta L_c = \int_{\partial B_c}\delta dl \quad , \quad \delta dl = (\delta L_{11}n_2^2 + \delta L_{22}n_1^2 - (\delta L_{12} + \delta L_{21})n_1 n_2))dl$$

The problem is modeled as a two-dimensional plane stress problem. By symmetry, only one quarter of the plate is analyzed. The material properties of the plate are: $E = 5.88 \times 10^3$ MPa (Young's modulus), $\nu = 0.3$ (Poisson ratio). The geometry and boundary conditions of the problem are shown in Fig. 5. The applied load is taken as $\sigma_{11} = 1.0$ Mpa, $\sigma_{22} = 0.75$ Mpa. The analytical solution of this problem [29] shows that the optimum shape of the hole is elliptic, with the semi-axes a and b given the ratio

$$\frac{b}{a} = \frac{\sigma_{22}}{\sigma_{11}} = 0.75$$

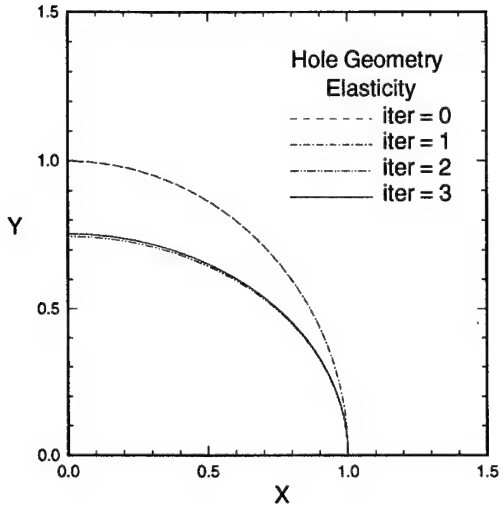

Figure 6: Convergence of hole geometry to optimum shape. One design parameter. Linear Elasticity.

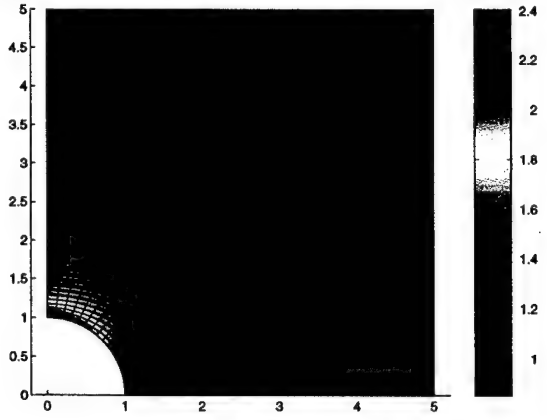

Figure 7: VonMises stress for initial hole geometry. One design parameter. Linear Elasticity

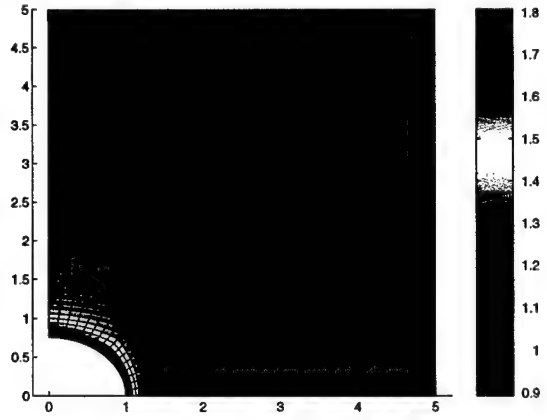

Figure 8: VonMises stress for optimized hole geometry. One design parameter. Linear Elasticity

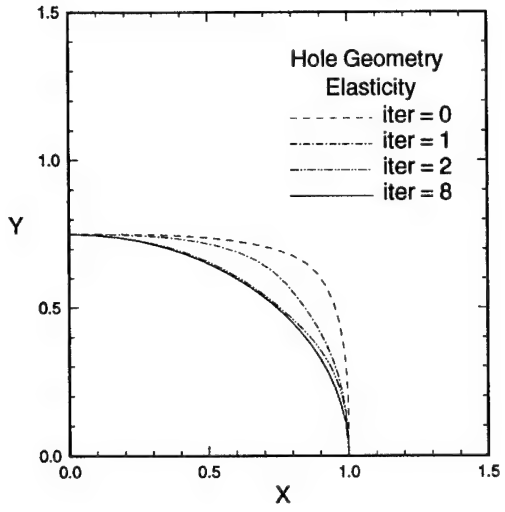

Figure 9: Convergence of hole geometry to optimum shape. Six design parameters. Linear Elasticity.

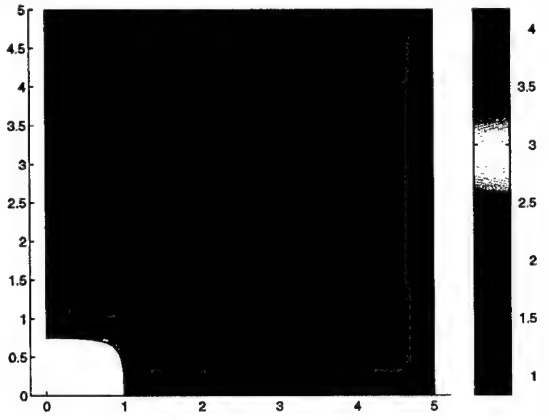

Figure 10: VonMises stress for initial hole geometry. Six design parameters. Linear Elasticity

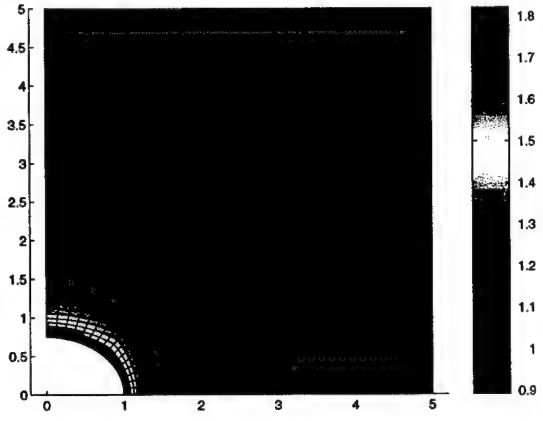

Figure 11: VonMises stress for optimized hole geometry. Six design parameters. Linear Elasticity

The PDESOLVE code for solving this example is similar to the one presented in the previous section and will not be shown here. Two cases are solved with this code. In the first case, the shape of the cutout is assumed to be elliptic and modeled as $x_1 = a\cos\theta$, $x_2 = b\sin\theta$. The geometry of the problem is defined by $L = 5$ m, $a = 1$ m and b is the design parameter. Numerical results are presented in Figs. 6 - 8. Fig. 6 shows the different geometries of the hole computed during the iterative solution procedure. The starting geometry was a circle with $b = 1$ m ($\phi = 1.5 \times 10^{-1}$), and the final (optimum) geometry is an ellipse with $b = 0.755$ m ($\phi = 4.7 \times 10^{-6}$). The optimum design was obtained in 3 iterations. Figures 7 and 8 shows the Von Mises stress distribution throughout the plate. Note that at the hole boundary, the Von Mises and tangential stresses are the same. Fig. 7 is the stress distribution prior to optimization ($b = 1$) and Fig. 8 is the stress distribution after optimization ($b = 0.755$). It is observed that for the optimum value of b the Von Mises stress is distributed uniformly along the hole.

For the second case, we take $L = 1$ m, $a = 1$ m, the optimum analytical value $b = 0.75$ m, and start with an arbitrary hole geometry represented by two pieces of third order Bezier curves (14 control points). The selected geometry of the hole, the symmetry conditions of the plate and the C^1 continuity requirement imposed at the joint point of the two curves leave us with 6 independent control points which are taken as the shape design parameters. The initial guess for these parameters defines the hole geometry presented in Fig. 10. This figure also shows the distribution of Von Mises stress throughout the plate for this initial geometry. Fig. 9 shows the different geometries of the hole computed during the iterative solution procedure. It is noted that the optimum shape of the hole (ellipse or close to an ellipse) is obtained after 8 iterations (the objective function reduces from $\phi = 5.4 \times 10^{-1}$ for the initial geometry to $\phi = 8.1 \times 10^{-5}$ for the optimum geometry). Fig. 11 shows the Von Mises stress distribution for the optimum hole shape. It is observed again that for this optimum shape, the stress is distributed uniformly along the hole.

4.3 Design Optimization in Small Deformation Plasticity

4.3.1 Problem Formulation

For the general formulation, we use a unified elasto-viscoplastic model with a single scalar internal variable (hardness) to characterize the plastic deformation of a material. The specific (rate) equations are given by the small strain version of the constitutive equations (14)-(17),

$$\dot{\boldsymbol{\sigma}} = \boldsymbol{C}:(\dot{\boldsymbol{\epsilon}} - \dot{\boldsymbol{\epsilon}}^p) \tag{69}$$

$$\dot{\boldsymbol{\epsilon}}^p = \sqrt{\frac{3}{2}}\dot{\bar{\epsilon}}^p \mathbf{N} \tag{70}$$

$$\dot{\bar{\epsilon}}^p = f(\bar{\sigma}, s) \tag{71}$$

$$\dot{s} = g(\bar{\sigma}, s) \tag{72}$$

where $\mathbf{N} = \frac{\boldsymbol{\sigma}'}{\|\boldsymbol{\sigma}'\|}$, $\bar{\sigma} = \sqrt{\frac{3}{2}} \parallel \boldsymbol{\sigma}' \parallel$ and $\boldsymbol{\sigma}' = \boldsymbol{\sigma} - \frac{1}{3}tr(\boldsymbol{\sigma})\mathbf{I}$. Here, $\dot{\boldsymbol{\epsilon}}$ and $\dot{\boldsymbol{\epsilon}}^p$ denote the total strain rate and its plastic component, respectively. These equations together with eqs.(61)-(63) (equilibrium equations and boundary conditions) define the direct problem for small deformation plasticity. The numerical solution of these nonlinear equations is carried out using an incremental, displacement-based approach. For this purpose, the specified time interval in divided into a certain number of steps Δt, with a typical time step defined by (t_N, t_{N+1}) where $t_{N+1} = t_N + \Delta t$. Assuming that the problem has been solved up to time t_N, the incremental solution procedure is focussed on computing the corresponding field variables at t_{N+1}. As before, this solution procedure involves iterations at both the equilibrium and the constitutive level. For the equilibrium iterations, we write eqs.(61)-(63) at t_{N+1} as

$$\boldsymbol{\nabla}\cdot\boldsymbol{\sigma}_{N+1} + \mathbf{f}_{N+1} = \mathbf{0} \quad \text{in } V \tag{73}$$

$$\boldsymbol{\sigma}_{N+1}\cdot\mathbf{n} = \bar{\mathbf{t}}_{N+1} \quad \text{on } A_\sigma \tag{74}$$

$$\mathbf{u}_{N+1} = \bar{\mathbf{u}}_{N+1} \quad \text{on } A_u \tag{75}$$

For algorithm development, it is convenient to express the stress and displacement at t_{N+1} as: $\boldsymbol{\sigma}_{N+1} = \boldsymbol{\sigma}^{(i)} + \Delta\boldsymbol{\sigma}$ and $\mathbf{u}_{N+1} = \mathbf{u}^{(i)} + \Delta\mathbf{u}$, where $\Delta\boldsymbol{\sigma}$ and $\Delta\mathbf{u}$ are the incremental stress and displacement, respectively. Here, the i-superscripted quantities denotes values of $\boldsymbol{\sigma}$ and $\mathbf{u}$ at the i-th iteration, with $\boldsymbol{\sigma}^{(0)} = \boldsymbol{\sigma}_N$ and

$\mathbf{u}^{(0)} = \mathbf{u}_N$. Using these expressions for $\boldsymbol{\sigma}_{N+1}$ and $\mathbf{u}_{N+1}$, we can write eqs(73)-(75) as

$$\begin{aligned} \boldsymbol{\nabla}\cdot(\Delta\boldsymbol{\sigma}) &= -\mathbf{f}_{N+1} - \boldsymbol{\nabla}\cdot\boldsymbol{\sigma}^{(i)} && (76)\\ \Delta\boldsymbol{\sigma}\cdot\mathbf{n} &= \bar{\mathbf{t}}_{N+1} - \boldsymbol{\sigma}^{(i)} && (77)\\ \Delta\mathbf{u} &= \bar{\mathbf{u}}_{N+1} - \mathbf{u}^{(i)} && (78) \end{aligned}$$

In these equation, the stress increment $\Delta\boldsymbol{\sigma}$ is related to the strain increment $\Delta\boldsymbol{\epsilon}$ using the elasto-viscoplastic constitutive matrix $\boldsymbol{\mathcal{L}}$

$$\Delta\boldsymbol{\sigma} = \boldsymbol{\mathcal{L}}:\Delta\boldsymbol{\epsilon} \qquad (79)$$

where

$$\begin{aligned} \boldsymbol{\mathcal{L}} &= 2\hat{\mu}\boldsymbol{\mathcal{I}} + (\kappa - \frac{2}{3}\hat{\mu})\mathbf{I}\otimes\mathbf{I} - 2\mu(\eta - c)\mathbf{N}^{(i)}\otimes\mathbf{N}^{(i)} && (80)\\ \Delta\boldsymbol{\epsilon} &= \frac{1}{2}[\boldsymbol{\nabla}(\Delta\mathbf{u}) + (\boldsymbol{\nabla}(\Delta\mathbf{u}))^T] && (81) \end{aligned}$$

The equilibrium iterations are then accomplished using eqs.(76)-(81) to obtain the incremental displacement $\Delta\mathbf{u}$. On the other hand, the constitutive iterations are carried out using the integrated form of the constitutive equations, eqs.(69)-(73), to update the stress $\boldsymbol{\sigma}_{N+1}$ and material state (hardness) s_{N+1}. Details of the constitutive integration procedure are presented in [22]

For the sensitivity problem, the governing equations are given by

$$\begin{aligned} \boldsymbol{\nabla}\cdot\delta\boldsymbol{\sigma}_{N+1} - \boldsymbol{\nabla}\boldsymbol{\sigma}_{N+1}:\delta\mathbf{L}^{\mathrm{T}} + \delta\mathbf{f}_{N+1} &= \mathbf{0} \quad \text{in } V && (82)\\ (\delta\boldsymbol{\sigma}_{N+1} - \boldsymbol{\sigma}_{N+1}\cdot\delta\mathbf{L}^{\mathrm{T}})\cdot\mathbf{n} + (\delta\mathbf{L}:\mathbf{n}\otimes\mathbf{n})\bar{\mathbf{t}}_{N+1} &= \delta\bar{\mathbf{t}}_{N+1} \quad \text{on } A_\sigma && (83)\\ \delta\mathbf{u}_{N+1} &= \delta\bar{\mathbf{u}}_{N+1} \quad \text{on } A_u && (84) \end{aligned}$$

together with the sensitivity equations for the constitutive model. These last equations relates the stress variation $\delta\boldsymbol{\sigma}_{N+1}$ to the strain variation $\delta\Delta\boldsymbol{\epsilon}$ and are derived from the integrated form of the eqs.(69)-(73). Omitting details of this derivation, these equations are

$$\begin{aligned} \delta\boldsymbol{\sigma}_{N+1} &= \boldsymbol{\mathcal{L}}_{N+1}:\delta\Delta\boldsymbol{\epsilon} + \boldsymbol{\mathcal{A}}_{N+1}:\delta\boldsymbol{\sigma}_N + \delta\alpha_N\mathbf{N}_{N+1} && (85)\\ \delta s_{N+1} &= \frac{c}{b_2}(b_1\delta\bar{\sigma}* + a_1\delta s_N) && (86) \end{aligned}$$

where $\delta\boldsymbol{\sigma}_N$ and δs_N represent the variation of the Cauchy stress and the hardness at time t_N, respectively, and

$$\begin{aligned} \boldsymbol{\mathcal{A}}_{N+1} &= \eta\boldsymbol{\mathcal{I}} + \frac{1-\eta}{3}\mathbf{I}\otimes\mathbf{I} - (\eta - c)\mathbf{N}_{N+1}\otimes\mathbf{N}_{N+1} && (87)\\ \delta\alpha_N &= -\sqrt{\frac{2}{3}}\frac{c\,a_2}{b_2}\delta s_N && (88)\\ \delta\Delta\boldsymbol{\epsilon} &= \frac{1}{2}[\boldsymbol{\nabla}\delta\Delta\mathbf{u} + (\boldsymbol{\nabla}\delta\Delta\mathbf{u})^T - (\boldsymbol{\nabla}(\Delta\mathbf{u})\cdot\delta\mathbf{L} + (\boldsymbol{\nabla}(\Delta\mathbf{u})\cdot\delta\mathbf{L})^T)] && (89) \end{aligned}$$

Equations (82)-(89) are solved for $\delta\Delta\mathbf{u}$ at each time step once the solution for the direct problem had been obtained. Note that the sensitivity equations are linear in $\delta\Delta\mathbf{u}$

4.3.2 Implementation of 2-D Small Deformation Plasticity in PDESolve

The previous equations have been simplified to 2-Dimensional plane stress and plane strain cases, and implemented using PDESolve. The module for the solution of the direct plane strain problem is presented below. The prescribed boundary conditions shown in this program correspond to those of a thick-wall cylinder under a prescribed velocity at its inner surface. The analytic solution for the radial stress σ_{rr} is given in [26] for a rigid, rate-independent plastic material. This solution and its corresponding sensitivity $\delta\sigma_{rr}$ to the inner radius a are compared here with the values predicted by the PDESolve program. The particular viscoplastic constitutive model used in this analysis is a power law of the form

$$\bar{\sigma} = \sigma_0(1 + \frac{\epsilon^p}{\epsilon_0})^n(1 + \frac{\dot{\epsilon}^p}{\dot{\epsilon}_0})^m \qquad (90)$$

where $\bar{\sigma}$ is the equivalent (Von Mises) stress, ϵ^p is the equivalent plastic strain, $\dot{\epsilon}^p$ is the equivalent plastic strain rate, and $\sigma_0, \epsilon_0, \dot{\epsilon}_0, n, m$ are material parameters. In this example, we use: $\sigma_0 = 50$ MPa, $\epsilon_0 = 0.002$, $\dot{\epsilon}_0 = 0.002s^{-1}$, $n = 0$, $m = 0$. The elastic constants are $E = 25 \times 10^3$ MPa, $\nu = 0.3$. The problem is solved incrementally, using 24 increments with a constant time step of $\Delta t = 0.5s$. The prescribed velocity at the inner surface is $\frac{\dot{u}}{a\dot{\epsilon}_0} = 1.0$. Figure 12 presents the nondimensional radial displacement $\frac{u}{a\epsilon_0}$ versus pressure ratio $\frac{p}{\sigma_0}$ $(p = -\sigma_{rr})$ for a material point located at the inner surface of the cylinder. The numerical solution was computed using both second (D2) and fourth (D4) order of accuracy in the representation of the finite difference operators $\frac{\partial}{\partial x_i}, i = 1, 2$. The results are in good agreement with the analytical solution. Figure 13 shows $\frac{u}{a\epsilon_0}$ versus $\frac{\delta\sigma_{rr}}{\sigma_0}$ for the same material point. As before, the symbols S2 and S4 indicate the order of accuracy of the finite difference stencil used to solve the sensitivity problem. It is important to note the effect of the finite difference representation on the computed sensitivities, in particular in the transition region from elastic to viscoplastic behavior. As deformation proceeds, the calculated sensitivities have the same trend as the analytical ones, obtaining for this problem a better match with the analytical solution using the fourth order precision of the finite difference representation.

```
void Small_Plasticity_Direct()
{
// Domain and Finite Difference Mesh
   SpatialDomain computationalDomain = SpatialDomain(0,1)*SpatialDomain(0,1);
   Function fwdMap(2,Tensor(2,1),fwdMapFunc,Function(2,Tensor(2,2),DfwdMapFunc) );
   Function invMap(2,Tensor(2,1),invMapFunc);
   SpatialDomain physicalDomain(computationalDomain, Map(fwdMap, invMap));
   FDMesh grid = FDMesh(physicalDomain, FDMeshSpec(nX) * FDMeshSpec(nY));

// Differential Operators
   DiffOp Dx(grid,1,0,4);
   DiffOp Dy(grid,1,1,4);

// Function Definitions
   Function U(2,Tensor(2,1));                          /*displacement*/
   Function dU(2,Tensor(2,1));                         /*accumulated incremental displacement*/
   Function T(2,Tensor(4,1));                          /*Cauchy stress*/
   Function EE(2,Tensor(4,1));                         /*elastic strain*/
   Function EP(2,Tensor(4,1));                         /*plastic strain*/
   Function EQP(2,Tensor(1,1));                        /*equivalent plastic strain*/
   Function Tt(2,Tensor(4,1));                         /*trial Cauchy stress*/
   Function EEt(2,Tensor(4,1));                        /*trial elastic strain*/
   Function EPt(2,Tensor(4,1));                        /*trial plastic strain*/
   Function EQPt(2,Tensor(1,1));                       /*trial equivalent plastic strain*/
   Function J4x4(2,Tensor(4,2));                       /*Consistent Tangent Operator*/
   Function F(2,Tensor(2,1), bodyForceFunc);           /*body force*/
   Function velBar(2,Tensor(2,1),BCVelocFuncMag);      /*prescribed velocity*/
   Function UBar(2,Tensor(2,1),zeroVectorFunc);        /*accumulated prescribed displacement*/
   Function strikeVect4To3(2,Tensor(3,1),strikeFuncVect4To3);   /*vector 4x1 -> vectpr 3x1*/
   Function strikeMatr4To3(2,Tensor(3,2),strikeFuncMatr4To3);   /*matrix 4x4 -> matrix 3x3*/

// Problem Solution
   U = zeroVec;
   T = zeroTen;
   EE = zeroTen;
   EP = zeroTen;
   EQP = zeroSca;
   for (incr = 0; incr < nIncrs; ++incr) {
      dU = zeroVec;
      UBar = UBar + dTime * velBar;
      int itersEquil = 0;
      do {
          if (itersEquil > maxItersEquil) throwRunTimeError("Equilibrium did not converge");
```

```
//        Constitutive Model Integration and Consistent Tangent Operator
          Tt = T;
          EEt = EE;
          EPt = EP;
          EQPt = EQP;
          Expr diffOperDirect = List(List(Dx, 0.0, Dy), List(0.0, Dy, Dx));
          Function dE3 = transpose(diffOperDirect)*dU;
          Function dE4 = List(dE3[0], dE3[1], 0.0, dE3[2]);
          Function listInMat = List(dE4, Tt, EEt, EPt, EQPt);
          Function listOutMat = MaterialConstIntegOp(listInMat);
          Tt = listOutMat[0];
          EEt = listOutMat[1];
          EPt = listOutMat[2];
          EQPt = listOutMat[3];
          J4x4 = listOutMat[4];
          Function J3x3 = strikeMatr4To3(J4x4);
          Function Ttt = strikeVect4To3(Tt);

//        Compute the displacement increment ddU
          Function ddU(2, Tensor(2, 1));
          ddU.regardAsUnknown();
          Expr ddE = transpose(diffOperDirect)*ddU;
          Expr dT = J3x3 * ddE;
          Expr equilEqns = diffOperDirect*(dT + Ttt) + F;
          BC bc = BC(List((dT[0]+Ttt[0])*Normal(0) + (dT[2]+Ttt[2])*Normal(1) == 0.0,
                          U[1]+ddU[1] == 0.0),
                     List(U[0] + ddU[0] == 0.0,
                          (dT[2]+Ttt[2])*Normal(0) + (dT[1]+Ttt[1])*Normal(1) == 0.0))
                * BC(List(U[0] + ddU[0] == UBar[0],
                          U[1] + ddU[1] == UBar[1]),
                     List((dT[0]+Ttt[0])*Normal(0) + (dT[2]+Ttt[2])*Normal(1) == 0.0,
                          (dT[2]+Ttt[2])*Normal(0) + (dT[1]+Ttt[1])*Normal(1) == 0.0));
          DiscPDO plastDirect(equilEqns, grid, ddU, bc);
          ddU = plastDirect.solve();

//        Update U and dU; Check Convergence in ddU
          dU = dU + ddU;
          U = U + ddU;
          itersEquil++;
          err1 = l2Norm(ddU);
          err2 = l2Norm(U);
      }
      while ((err1 / err2) > tolerEquil);

//    Update Variables
      T = Tt;
      EE = EEt;
      EP = EPt;
      EQP = EQPt;
}
```

4.3.3 Application of PDESolve to Optimization Problems in Small Deformation Plasticity

In this section, we solve the shape optimization of elliptic cutouts in thin elasto-viscoplastic plates subjected to constant ratios of far field stresses (see Fig. 5). Again, we want to compute the optimum shape of the cutout for a uniform tangential stress distribution along it. Hence, the objective function and its sensitivity are the same as the ones presented before. In this example, the deformation of the plate, modeled as a plane stress problem, is driven by the time dependent loading: $\sigma_{xx}(t) = 16.0 + 8t$ MPa, $\sigma_{yy}(t) = 0.75\sigma_{xx}(t)$ MPa, where t

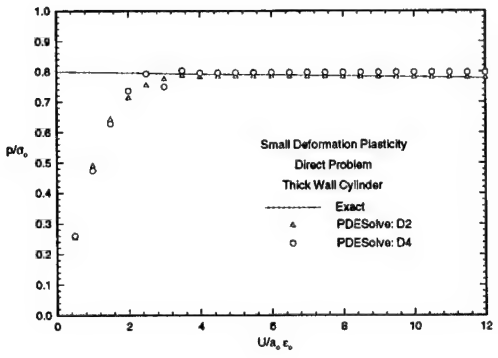

Figure 12: Radial stress ($\sigma_{rr} = -p$) at $r = a$ of thick wall cylinder. Small Deformation Plasticity

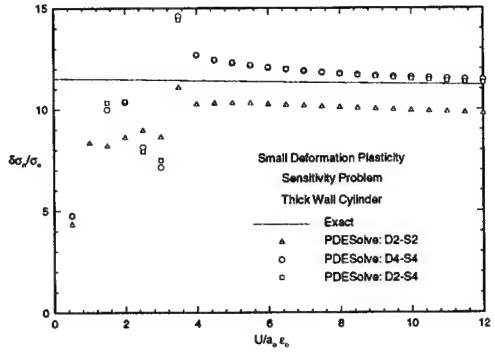

Figure 13: Sensitivity $\delta\sigma_{rr}$ at $r = a$ of thick wall cylinder. Small Deformation Plasticity

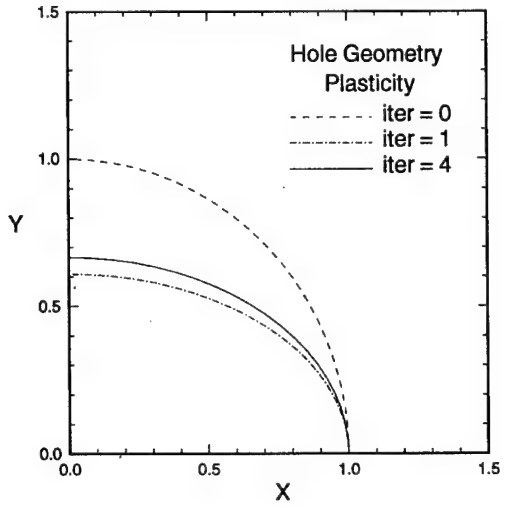

Figure 14: Convergence of hole geometry to optimum shape. Small Deformation Plasticity.

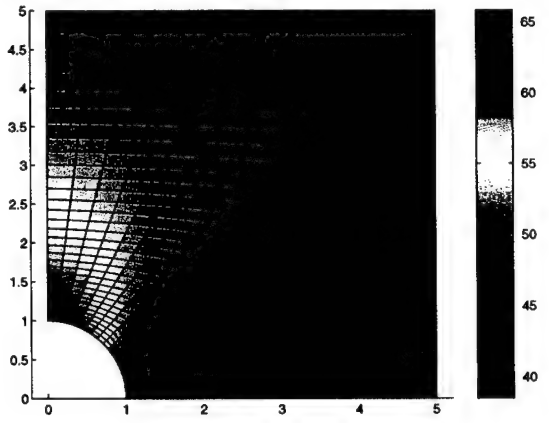

Figure 15: VonMises stress for initial hole geometry. Small Deformation Plasticity

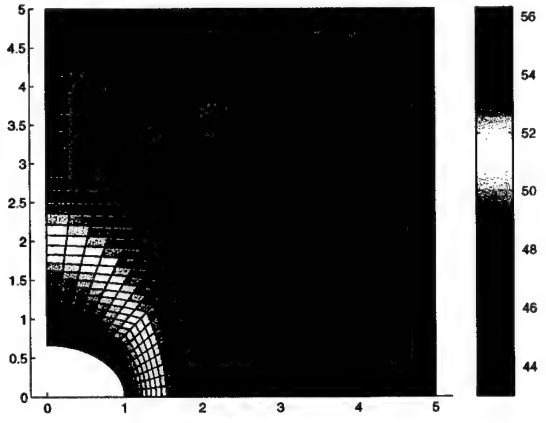

Figure 16: VonMises stress for optimized hole geometry. Small Deformation Plasticity

is time. We assume an elliptic geometry of the hole ($x_1 = a\cos\theta$, $x_2 = b\sin\theta$) and take $L = 5$ m, $a = 1$ m, with b being the design parameter. The material parameters used are $\sigma_0 = 50$ MPa, $\epsilon_0 = 0.002$, $\dot{\epsilon}_0 = 0.002s^{-1}$, $n = 0.1$, $m = 0.1$, $E = 25 \times 10^3$, $\nu = 0.3$. The plasticity solution of the direct and sensitivity problems is carried out incrementally for a prescribed time interval. Specifically, the problem is solved using 25 increments with a constant time step of $\Delta t = 0.16s$ to give a total time of $4s$. We use a second order accuracy of the finite difference representation for solving both the direct and the sensitivity problems. Numerical results are presented in Figs. 14 - 16. Fig. 14 shows the different hole geometries computed during the optimization process. The starting geometry was a circle with $b = 1$ m ($\phi = 59.7$), and the final (optimum) geometry is an ellipse with $b = 0.667$ m ($\phi = 7.6 \times 10^{-3}$). The optimum elasto-viscoplastic design is obtained after 4 iterations. Note here that the stress distribution along the cutout for the elasticity analytical solution $b = 0.75$ m is not longer the optimal when inelastic deformation occurs. Hence, the optimum shape of the cutout for the new stress pattern is given by $b = 0.667$ m. Figs. 15 and 16 shows the Von Mises stress distribution throughout the plate. Fig. 15 is the stress distribution prior to optimization ($b = 1$) and Fig. 16 is the stress distribution after optimization ($b = 0.667$). We observe that for the optimum value of b the Von Mises stress distribution is uniform along the hole.

5 PDESolve AApplied to Shape Optimization of One-Step Open Die Forging

5.1 Implementation of 2-D Large Deformation Plasticity in PDESolve

Before applying the general optimization-based design methodology to the forging problem, in this section we validate the 2-D plane strain implementation of large deformation plasticity using PDESolve. The particular module for solving the large-deformation direct problem is presented below. With this code, we solve the direct and sensitivities problems of the large deformation of a thick-wall cylinder subjected to an applied velocity at its inner surface. As before, the analytical solution, given in [26], is used here to compare with the prediction of PDESolve. We use the viscoplastic model given by eq.(90) with the material parameters: $\sigma_0 = 50$ MPa, $\epsilon_0 = 0.002$, $\dot{\epsilon}_0 = 0.002s^{-1}$, $n = 0$, $m = 0$. Also, $E = 25 \times 10^3$ MPa and $\nu = 0.3$. The problem is solved incrementally, using 25 increments with a constant time step of $\Delta t = 20.0s$ to give a total time of $500s$. The prescribed velocity at the inner surface is again $\frac{\dot{u}}{a\dot{\epsilon}_1} = 1.0$. The plots $\frac{u}{a\epsilon_0}$ vs $\frac{p}{\sigma_0}$ and $\frac{u}{a\epsilon_0}$ vs $\frac{\delta\sigma_{rr}}{\sigma_0}$ for a material point at the inner surface of the cylinder are shown in Figs. 17 and 18, respectively. We observe that the numerical solution of the direct problem is in good agreement with the analytical one. The computed sensitivity coeficients, on the other hand, have the same trends as the analytical ones, with a maximum error of approximately 25%.

```
void Large_Plasticity_Direct()
{
// Domain and Finite Difference Mesh
   SpatialDomain computationalDomain = SpatialDomain(0,1)*SpatialDomain(0,1);
   Function fwdMap(2,Tensor(2,1),fwdMapFunc,Function(2,Tensor(2,2),DfwdMapFunc));
   Function invMap(2,Tensor(2,1),invMapFunc);
   SpatialDomain undeformedDomain(computationalDomain, Map(fwdMap,invMap));
   FDMesh undeformedMesh = FDMesh(undeformedDomain, FDMeshSpec(nX)*FDMeshSpec(nY));
   FDDiscretize undeformedBody(undeformedMesh);        /*initial configuration, B_0*/
   Function X(2,Tensor(2,1),coordFunc);
   Function Xini = undeformedBody.discretize(X);       /*initial coordinates, x_0*/
   Function Xref = undeformedBody.discretize(X);       /*reference coordinates, x_N*/
   Function Xcur(2,Tensor(2,1));                       /*current coordinates, x_(N+1)*/

// Differential Operators
   DiffOp Dx(1, 0, 4);
   DiffOp Dy(1, 1, 4);
   Expr Del = List(Dx, Dy);

// Function Definitions
   Function U(2,Tensor(2,1));                          /*displacement*/
   Function Ut(2,Tensor(2,1));                         /*trial displacement*/
```

```
   Function dU(2,Tensor(2,1));                    /*accumulated incremental displacement*/
   Function T(2,Tensor(4,1));                     /*Cauchy stress*/
   Function EE(2,Tensor(4,1));                    /*elastic strain*/
   Function EP(2,Tensor(4,1));                    /*plastic strain*/
   Function EQP(2,Tensor(1,1));                   /*equivalent plastic strain*/
   Function DEQPt(2,Tensor(1,1));                 /*incremental EQP*/
   Function Tt(2,Tensor(4,1));                    /*trial Cauchy stress*/
   Function EEt(2,Tensor(4,1));                   /*trial elastic strain*/
   Function EPt(2,Tensor(4,1));                   /*trial plastic strain*/
   Function EQPt(2,Tensor(1,1));                  /*trial equivalent plastic strain*/
   Function J4x4(2,Tensor(4,2));                  /*consisten tangent operator*/
   Function FN(2,Tensor(2,2));                    /*deformation gradient F_N*/
   Function FN1(2,Tensor(2,2));                   /*deformation gradient F_(N+1)=F_u*/
   Function DF(2,Tensor(2,2));                    /*relative deformation gradient F*/
   Function dF(2,Tensor(4,1));                    /* F in vector form*/
   Function J5x5(2,Tensor(5,2));                  /*consistent tangent operator*/
   Function B(2,Tensor(2,1),bodyForceFunc);       /*body force, B=0 in this example*/
   Function velBar(2,Tensor(2,1),BCVelocFuncMag); /*prescribed velocity*/
   Function UBar(2,Tensor(2,1),zeroVectorFunc);   /*accumulated prescribed displacement*/

// Problem Solution
   U = undeformedBody.discretize(zeroVec);
   T = undeformedBody.discretize(zeroTen);
   EE = undeformedBody.discretize(zeroTen);
   EP = undeformedBody.discretize(zeroTen);
   EQP = undeformedBody.discretize(zeroSca);
   dTimeN = dt[0];   /*This module includes an automatic time stepping scheme*/
   time = 0.0;       /*Not used in this problem*/
   incr = 0;
   counter = 0;
   do {
      incr = incr + 1;
//    Define reference configuration and setup equilibrium iterations
      SpatialDomain referenceDomain(computationalDomain, Map(Xref));
      FDMesh referenceMesh = FDMesh(referenceDomain, FDMeshSpec(nX) * FDMeshSpec(nY));
      FDDiscretize referenceBody(referenceMesh);
      FN = transpose(outerProduct(Del, Xref));
      Tn = undeformedBody.discretize(T);
      redoEquilSol: ;
      convergeEquilSol = true;
      dTime = dTimeN;
      UBarTr = UBar + dTime * velBar;
      dU = referenceBody.discretize(zeroVec);
      Xcur = undeformedBody.discretize(Xref);
      Ut = undeformedBody.discretize(U);
      timeTr = time + dTime;
      int itersEquil = 0;
      do {
          if (itersEquil > maxItersEquil)
             { convergeEquilSol = false;
               cout << " convergeEquilSol = FALSE for dTime = " << dTime << endl;
               cout << " ... will try dTimeN = 0.5*dTime " << endl;
               goto checkTimeStep; }

//        Constitutive Model Integration and Consistent Tangent Operator
          Tt = referenceBody.discretize(T);
          EEt = referenceBody.discretize(EE);
          EPt = referenceBody.discretize(EP);
          EQPt = referenceBody.discretize(EQP);
          FN1 = transpose(outerProduct(Del, Xcur));
```

```
        DF = FN1 * inverseMatxOp(FN);
        dF = List(DF[0][0],DF[0][1],DF[1][0],DF[1][1]);
        Function listInMat = List(dF,Tt,EEt,EPt,EQPt);
        Function listOutMat = MaterialConstIntegOp(listInMat);
        Tt = listOutMat[0];
        EEt = listOutMat[1];
        EPt = listOutMat[2];
        EQPt = listOutMat[3];
        J5x5 = listOutMat[4];
        listPlast = listOutMat[6];
        Function J4x4 = List(List(J5x5[0][0],J5x5[0][1],J5x5[0][2],J5x5[0][3]),
                             List(J5x5[1][0],J5x5[1][1],J5x5[1][2],J5x5[1][3]),
                             List(J5x5[2][0],J5x5[2][1],J5x5[2][2],J5x5[2][3]),
                             List(J5x5[3][0],J5x5[3][1],J5x5[3][2],J5x5[3][3]));
        Function matxT1 = List(List(Tt[0],0.0,0.0,Tt[0]),
                               List(Tt[1],0.0,0.0,Tt[1]),
                               List(Tt[2],0.0,0.0,Tt[2]),
                               List(Tt[3],0.0,0.0,Tt[3]));
        Function matxT2 = List( List(Tt[0],0.0,Tt[1],0.0),
                               List(0.0,(0.5*(Tt[0]-Tt[3])),(0.5*(Tt[0]+Tt[3])),Tt[1]),
                               List(Tt[2],(0.5*(Tt[0]+Tt[3])),(0.5*(-Tt[0]+Tt[3])),0.0),
                               List(0.0,Tt[2],0.0,Tt[3]));
        Function matxX = List(  List(1.0,0.0,0.0,0.0),
                               List(0.0,0.5,0.5,0.0),
                               List(0.0,0.5,0.5,0.0),
                               List(0.0,0.0,0.0,1.0) );
        Function Ttt = List(Tt[0],Tt[1],Tt[2],Tt[3]);

//      Define current condifuration and compute the displacement increment ddU
        SpatialDomain currentDomain(computationalDomain, Map(Xcur));
        FDMesh currentMesh = FDMesh(currentDomain, FDMeshSpec(nX) * FDMeshSpec(nY));
        FDDiscretize currentBody(currentMesh);
        Function ddU(2,Tensor(2,1));
        ddU.regardAsUnknown();
        Expr diffOperDirect = List(List(Dx,Dy,0.0,0.0),List(0.0,0.0,Dx,Dy));
        Expr ddF = transpose(diffOperDirect) * ddU;
        Expr dP = (J4x4*matxX - matxT2 + matxT1) * ddF;        /*I Piola-Kircchhof stress*/
        Expr equilEqns = diffOperDirect*(dP + Ttt);
        BC bc = BC(List(Ut[0] + ddU[0] == UBarTr[0],
                        Ut[1] + ddU[1] == UBarTr[1]),
                   List((dP[0] + Ttt[0])*Normal(0) + (dP[1] + Ttt[1])*Normal(1) == 0.0,
                        (dP[2] + Ttt[2])*Normal(0) + (dP[3] + Ttt[3])*Normal(1) == 0.0))
              * BC(List((dP[0] + Ttt[0])*Normal(0) + (dP[1] + Ttt[1])*Normal(1) == 0.0,
                        Ut[1] + ddU[1] == 0.0),
                   List(Ut[0] + ddU[0] == 0.0,
                        (dP[2]+Ttt[2])*Normal(0) + (dP[3]+Ttt[3])*Normal(1) == 0.0));
        DiscExpr plastDirect(currentMesh, equilEqns, bc, ddU);
        ddU = plastDirect.solve();

//      Update Ut and dU; Check Convergence in ddU
        Xcur = undeformedBody.discretize(Xcur + ddU);
        dU = referenceBody.discretize(dU + ddU);
        Ut = undeformedBody.discretize(Ut + ddU);
        itersEquil++;
        err1 = l2Norm(ddU);
        err2 = l2Norm(Ut);
    }
    while ((err1 / err2) > tolerEquil);

//  Compute new or next time step (For Automatic Time Stepping Scheme)
```

```
      checkTimeStep: ;
      DEQPt = referenceBody.discretize(listPlast);
      if (autoTime == 0)
         { dTimeN = dt[incr];
           rejectEquilSol = false;
           if (!convergeEquilSol)
                throwRunTimeError("No autoTime was prescribed : exit program");}
      else if (autoTime == 1)
         { timeControlFunc(convergeEquilSol, rejectEquilSol, dTime,
                           dTimeN, counter, DEQPt); }
      if (rejectEquilSol)
         { cout << " WARNING: Redoing Equilibrium Solution" << endl;
           goto redoEquilSol; }

//    Update Body's Configuration and Variables
      Xref = undeformedBody.discretize(Xcur);
      SpatialDomain currentDomain(computationalDomain, Map(Xref));
      FDMesh currentMesh = FDMesh(currentDomain, FDMeshSpec(nX) * FDMeshSpec(nY));
      FDDiscretize currentBody(currentMesh);
      time = timeTr;
      UBar = UBarTr;
      U = undeformedBody.discretize(Ut);
      T = currentBody.discretize(Tt);
      EE = currentBody.discretize(EEt);
      EP = currentBody.discretize(EPt);
      EQP = currentBody.discretize(EQPt);
}
```

5.2 Shape Optimization of One-Step Open Die Forging

In this section, we use the general methodology developed for optimization-based design of multi-step forging to solve the preform shape design of one-step forging (upsetting) where a rigid die of fixed geometry is used. For this particular case, the methodology is simplified to the following steps:

1. Guess the initial value of the design parameters $\boldsymbol{\beta}$.
2. Do i = 1, # steps (incremental analysis):
   (a) Solve direct problem for the field variables $(\mathbf{u}, \mathbf{P})$. Use the integrated form of the constitutive equations, eqs.(20)-(23), and the incremental equilibrium equations, eqs.(27)-(29).
   (b) Solve n sensitivity problems for the variations $(\delta\mathbf{u}, \delta\mathbf{P})$, where n is the number of design parameters. Use eqs.(35)-(37) together with eqs.(51)-(52).
3. Solve optimization problem to compute new values of $\boldsymbol{\beta}$. Here, the objective function Φ (see below) is minimized using gradient information $(\delta\Phi)$.
4. Check convergence of the objective function:
   If $\| \Phi \| \leq$ Tolerance $\Rightarrow$ exit
   If $\| \Phi \| >$ Tolerance $\Rightarrow$ go to step 2 with the new values of $\boldsymbol{\beta}$.

The specific problem to be solved is the design of the preform shape such that after the forging (upsetting) process, the final product has a minimum barreling (see fig. 19). Modeling the problem as a two-dimensional plane strain problem (only one quarter of the preform need to be considered), we can write the objective function as [2]

$$\Phi(\boldsymbol{\beta}) = \int_{\Gamma_f} (x_1 - r_0)^2 d\Gamma \tag{91}$$

where r_0 is a prescribed value, Γ_f is the free surface of the deformed workpiece after upsetting, and the coordinate $x_1 = \mathbf{x} \cdot \mathbf{e}_1$ locates a material point on this free surface. Here, $\mathbf{x} = \mathbf{x}_0 + \mathbf{u}$, and $\mathbf{e}_1$ is a unit vector

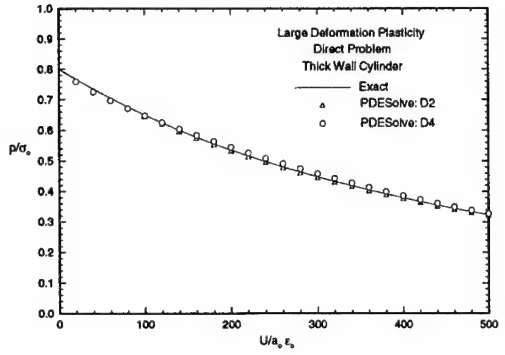

Figure 17: Radial stress ($\sigma_{rr} = -p$) at $r = a$ of thick wall cylinder. Large Deformation Plasticity

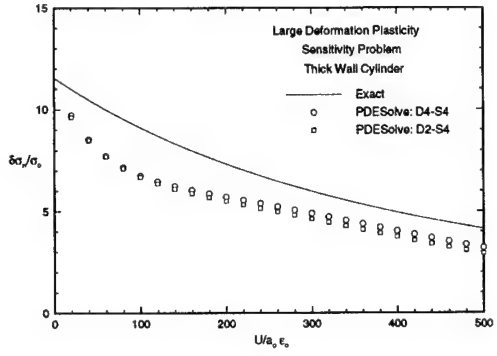

Figure 18: Sensitivity $\delta\sigma_{rr}$ at $r = a$ of thick wall cylinder. Large Deformation Plasticity

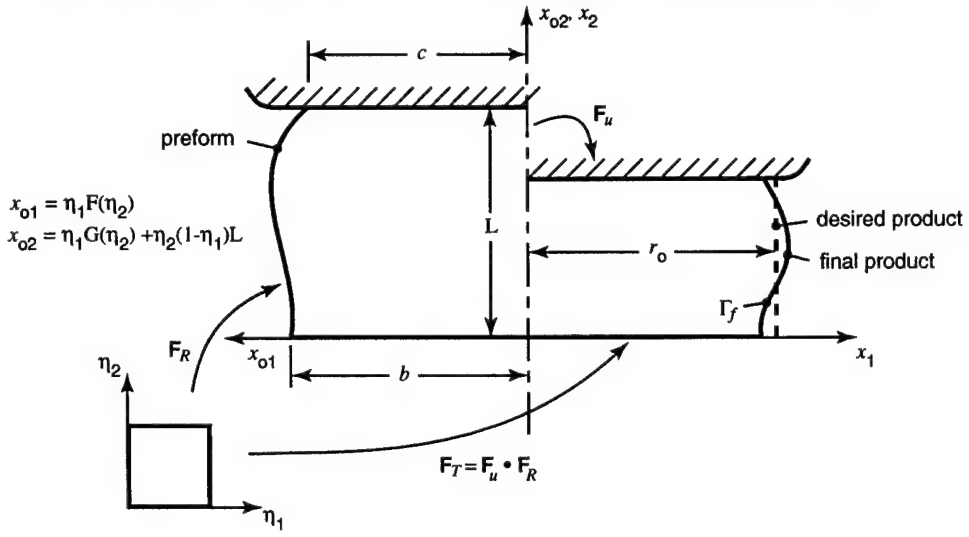

Figure 19: Preform design in one-step open die forging (upsetting).

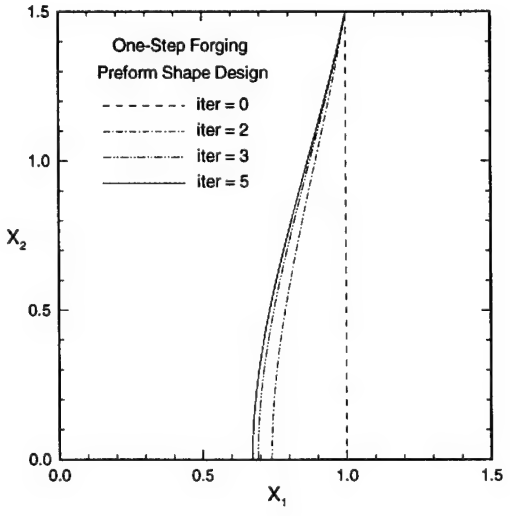

Figure 20: Convergence of preform free surface geometry to optimum shape. Four design parameters. One-step forging.

along the 1-direction. The variation of Φ is computed as

$$\delta\Phi = \int_{\Gamma_f} [2(x_1 - r_0)\delta x_1 + (x_1 - r_0)^2(tr(\delta\mathbf{L}) - \delta\mathbf{L}{:}\mathbf{n}\otimes\mathbf{n})]d\Gamma \tag{92}$$

where

$$\delta\mathbf{L} = \delta\mathbf{F}_T{\cdot}\mathbf{F}_T^{-1} = \delta\mathbf{F}_u{\cdot}\mathbf{F}_u^{-1} + \mathbf{F}_u{\cdot}\delta\mathbf{L}_0{\cdot}\mathbf{F}_u^{-1} \quad , \quad \delta\mathbf{L}_0 = \delta\mathbf{F}_R{\cdot}\mathbf{F}_R^{-1} \tag{93}$$

$$\delta\mathbf{x} = \delta\mathbf{x}_0 + \delta\mathbf{u} \quad , \quad \delta x_1 = \delta\mathbf{x}{\cdot}\mathbf{e}_1 \quad , \quad \delta x_{01} = \delta\mathbf{x}_{01}{\cdot}\mathbf{e}_1 \tag{94}$$

In these expressions, the field variables $(\mathbf{u}, \mathbf{F}_u)$ are computed from the solution of the direct problem while the variations $(\delta\mathbf{u}, \delta\mathbf{F}_u)$ are obtained from the sensitivity analysis of the problem. As shown in fig. 19, the preform geometry is mapped to the 2-D unit square using the bilinear transfinite mapping

$$\begin{aligned} x_{o1} &= \eta_1 F(\eta_2) \\ x_{o2} &= \eta_1 G(\eta_2) + \eta_2(1-\eta_1)L \end{aligned}$$

where the functions $F(\eta_2)$ and $G(\eta_2)$ define the geometry of the free boundary of the preform by a third order Bezier curve, i.e.,

$$\begin{aligned} F(\eta_2) &= (1-\eta_2)^3 a_0 + 3\eta_2(1-\eta_2)^2 a_1 + 3\eta_2^2(1-\eta_2)a_2 + \eta_2^3 a_3 \\ G(\eta_2) &= (1-\eta_2)^3 b_0 + 3\eta_2(1-\eta_2)^2 b_1 + 3\eta_2^2(1-\eta_2)b_2 + \eta_2^3 b_3 \end{aligned}$$

where $a_i, b_i, i = 0, 3$ are the control points of the Bezier curve with $F(0) = b$, $F(1) = c$, $G(0) = 0$, $G(1) = L$ (see Fig. 19). This mapping is used to compute the variation of the design velocity gradient $\delta\mathbf{L}_0$, which for the case of preform shape design, drives the sensitivity problem. For the numerical simulations, we will assume sticking friction at the workpiece-die interface. Also, we will use the viscoplastic relation defined by eq.(90) with the material parameters $\sigma_0 = 50$ MPa, $\epsilon_0 = 0.002$, $\dot{\epsilon}_0 = 0.002s^{-1}$, $n = 0.1$, $m = 0.1$, $E = 25 \times 10^3$ MPa, $\nu = 0.3$. The workpiece is deformed up to 20% upsetting with a compressive velocity of 12×10^{-3} mm/s using 25 increments with a time step size of $\Delta t = 1.0s$ (nominal strain rate of $0.009s^{-1}$)

The geometry of the preform and the symmetry conditions give $a_0 = a_1 = b$, $a_3 = c$, $b_0 = 0$ and $b_3 = L$. For this example, we fixed the values $c = 1$ mm, $L = 1.5$ mm, $r_0 = 1.0$ mm and take $a_0(= \beta_1)$,$a_2(= \beta_2)$,$b_1(= \beta_3)$ and $b_2(= \beta_4)$ as the four design parameters β_i. The initial guess for these parameters, given by: $\beta_1 = 1.0$, $\beta_2 = 1.0$, $\beta_3 = 0.5$, and $\beta_4 = 1.0$, defines the initial preform geometry shown in Fig. 21. Figure 22 shows the distribution of equivalent plastic strains throughout the deformed workpiece after 20% upsetting. We clearly see the barreling of the workpiece due to friction. For this deformed geometry, the objective function has the value $\phi = 8.3 \times 10^{-2}$. Figure 20 presents the different shapes of the preform free surface computed during the optimization process. The "optimum" shape is obtained after 5 iterations. The optimum design parameters are $\beta_0 = \beta_1 = 0.673$, $\beta_2 = 0.496$, $\beta_3 = 0.886$, $\beta_4 = 0.979$ giving a value for the objective function of $\phi = 4.3 \times 10^{-4}$. The initial preform geometry and the corresponding deformed workpiece are shown in Figs. 23 and 24. Although the deformed workpiece does not have a perfectly vertical free surface, it is observed that the geometry of the forged product is close to the one specified. We note here that this solution can be improved by using a better representation of the free surface as well as by mesh refinement.

6 A Roadmap for Constitutive Models to be Included in the Framework

The framework being developed in Phase I for optimization-based forging design uses a hypo-elastic constitutive model [22] for the material representation and simple contact/friction conditions (such as sticking friction) at the workpiece-die interface. Such simplified framework has been coded using our computational substrate PDESolve to solve preform design problems for one-step isothermal forging. For Phase II, this framework will be extended to incorporate constitutive laws that appropriately represent the thermo-elasto-viscoplastic-damage material behavior of typical metallic alloys of interest to the Air Force. A number of material models together with their numerical time integration procedure will be investigated. Appropriate models for representing the contact/friction conditions at the workpiece-die interface as well as the corresponding numerical

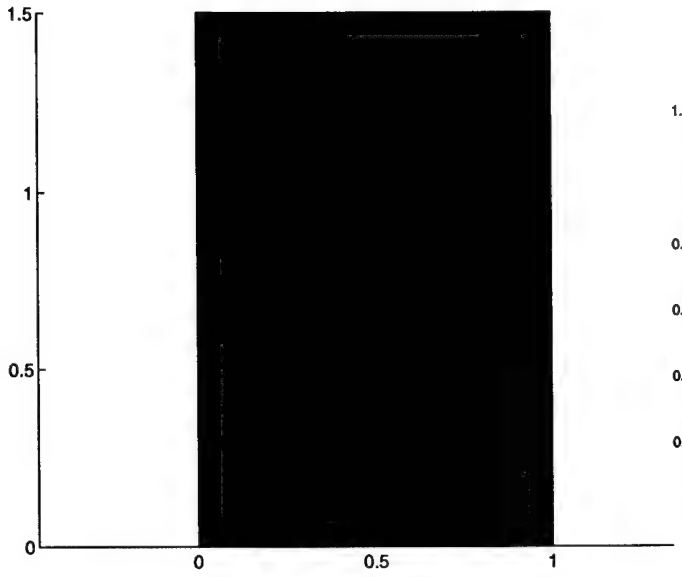

Figure 21: Preform geometry for initial guess of design parameters. One-step forging.

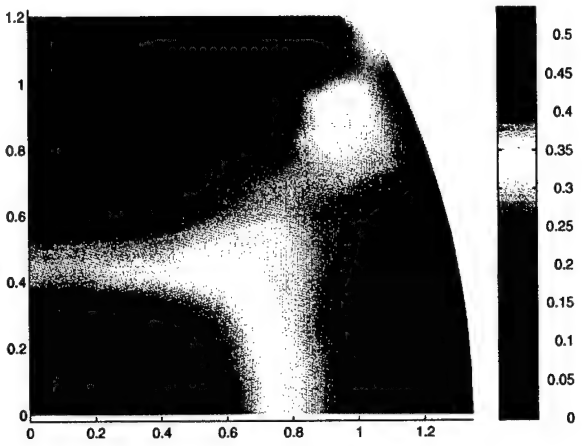

Figure 22: Deformed workpiece for guessed preform geometry after 20% upsetting. One-step forging

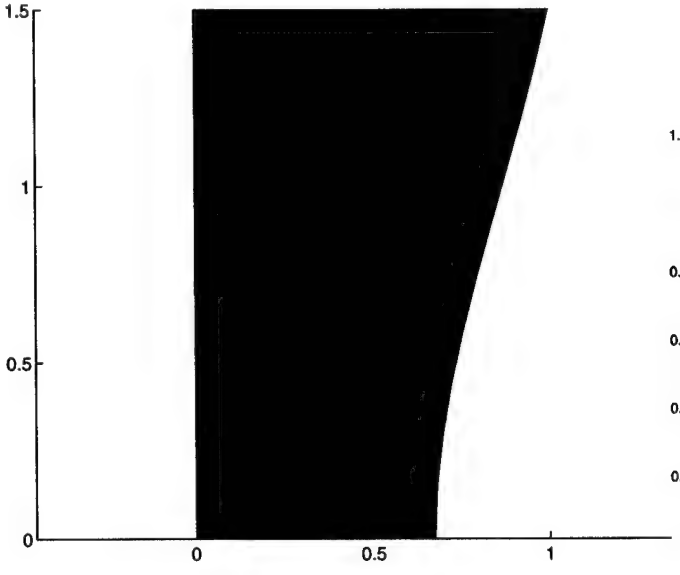

Figure 23: Preform geometry for optimum values of design parameters. One-step forging.

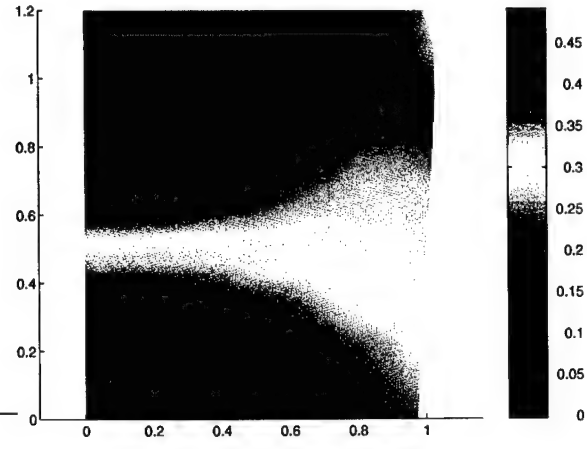

Figure 24: Deformed workpiece for optimum preform geometry after 20% upsetting. One-step forging.

algorithms will also be addressed. This extended constitutive description will give a more accurate description of the physics that controls the final product in a forging process.

One such aspect of the physics is the formation of voids or cavities in regions where high tensile stresses are produced. This phenomena leads to degradation of mechanical properties of the forged product, and hence, to its premature failure. We will incorporate damage models into the framework to predict zones where such conditions may occur and use this information in the optimization process to minimize the risk of void formation. Specific models based on phenomenological continuum approaches that account for isotropic/anisotropic damage descriptions will be investigated [13, 35].

Another aspect of the physics is the nonuniform distribution of plastic strain throughout the workpiece imparted during forging. This nonuniform plastic deformation produces residual stresses that may have important effects in the strength of the final product. Although plastic strains are unavoidable in large deformation processes, its distribution may be controlled to obtain favorable residual stress patterns, in particular during multi-step forging. Predicting residual stresses calls for constitutive models that account for the elastic response of the material. For this purpose, a number of elasto-viscoplastic models that use hyper- and hypo-elasticity together with internal variable representation will be investigated. In particular, constitutive laws with hardening rules that reflect microstructure evolution [4, 7, 24] will be studied.

Explicit modeling of microstructure features as represented by slip processes that operate during plastic flow as well as the reorientation of crystals (texture) comprising the metal will also be addressed. These features of the microstructure are introduced in numerical simulations of metal forming by polycrystal plasticity models [6, 19]. By inclusion of these models in the framework, we will able to predict texture evolution in the forged product, and hence, the optimization process can focus on producing an optimized geometry of the final product with desirable texture and mechanical properties.

Figure 9 summarizes the extended framework to be developed in Phase II for the optimization-based design of multi-step forging processes. This framework will be complemented with a methodology to be developed for deriving the corresponding sensitivity equations which are needed for the sensitivity analysis of the forging process. The specific constitutive models to be used in the framework will be chosen in coordination with the Air Force.

7 Conclusions

In this work (Phase I), we have developed a methodology for solving optimization-based design problems in linear and non-linear mechanics, along with the numerical tool using our computational substrate PDESolve. The solution of a number of optimization problems using linear elasticity and small- and large-deformation plasticity has shown the validity of the methodology as well as the flexibility of PDESolve for implementing complex tensorial differential equations. In particular, the solution of the preform shape design problem for a one-step forging process have shown the applicability of the present development to optimization-based forging design, a technological area of great interest to the Air Force.

In Phase II, we will built on the results obtained in Phase I to extend/improve the methodology and solution procedures to optimize the preform and die geometries in multi-step forging processes. In this case, the computational tool will be based on the finite element method, a version of PDESolve which is currently being developed.

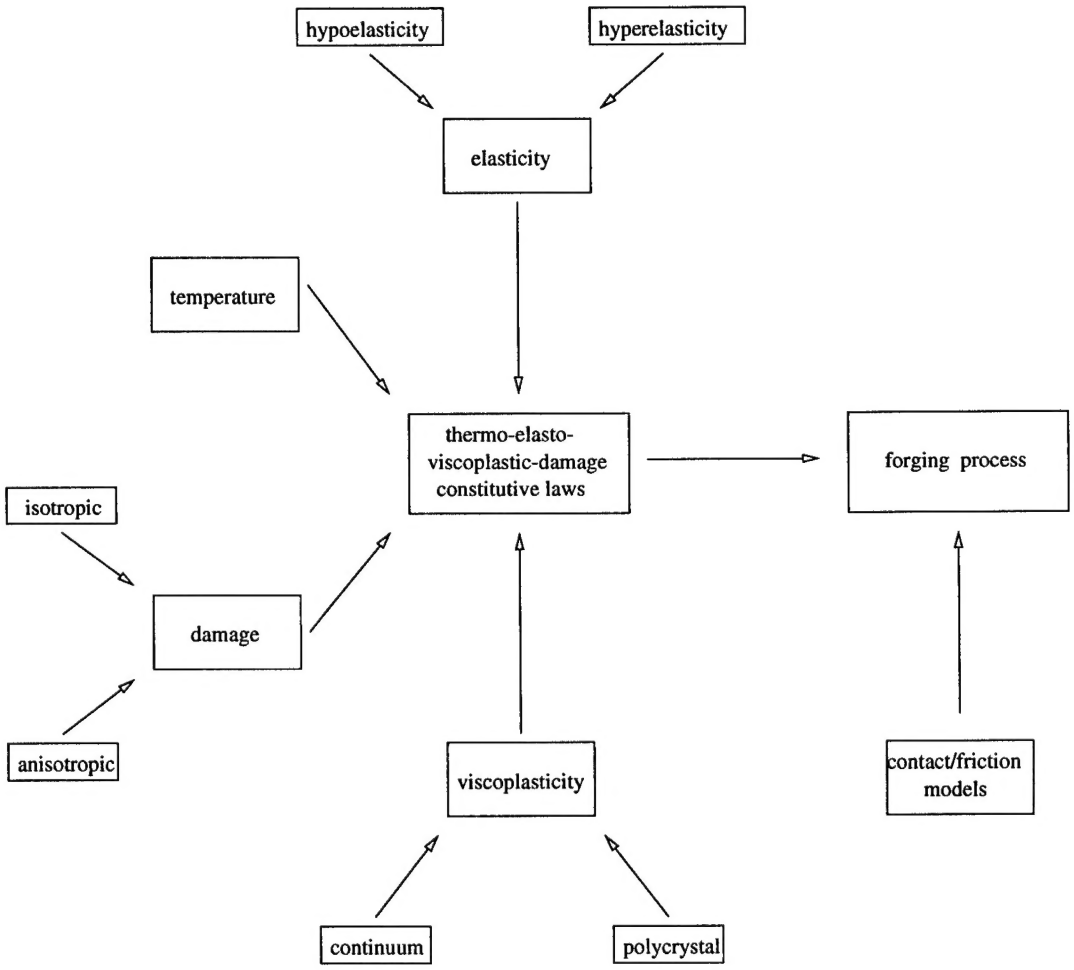

Figure 25: Schematic roadmap illustrating components of constitutive framework for modeling forging processes.

References

[1] J. S. Arora and J. B. Cardoso. A design sensitivity analysis principle and its implementation into adina. *Computers and Structures*, 32:691–705, 1989.

[2] S. Badrinarayanan and N. Zabaras. A sensitivity analysis for the optimal design of metal forming processes. *Computer Methods in Applied Mechanics and Engineering*, 129:319–348, 1996.

[3] S. Badrinarayanan, N. Zabaras, and A. Constantinescu. Preform design in metal forming. In S.-F. Shen and P. R. Dawson, editors, *Proceedings of the 5th International Conference On Numerical Methods in Industrial Forming Processes*, pages 533–538, June 1995.

[4] D.J. Bammann. Modeling the temperature and strain rate dependent large deformation of metals. *Applied Mechanics Reviews*, 43:S312–S319, 1990.

[5] A. J. Beaudoin, P.R. Dawson, K. K. Mathur, U. F. Kocks, and D. A. Korzekwa. Application of polycrystalline plasticity to sheet forming. *Computer Methods in Applied Mechanics and Engineering*, 117:49–70, 1994.

[6] A. J. Beaudoin, K. K. Mathur, P. R. Dawson, and G. C. Johnson. Three dimensional deformation process simulation with explicit use of polycrystalline plasticity models. *International Journal of Plasticity*, 9:833–860, 1993.

[7] J. L. Chaboche and G. Rousselier. On the plastic and viscoplastic constitutive equations - part i: Rules developed with internal variable concept. *Journal of Pressure Vessel Technology*, 105:153–158, 1983.

[8] M. Z. Cohn and A. S. Dinovitzer. Application of structural optimization. *Journal of Structural Engineering*, 120:617–650, 1994.

[9] K. Dems and R. T. Haftka. Two approaches to sensitivity analysis for shape variation of structures. *Mech. Struct. & Mach.*, 4:501–522, 1988-89.

[10] L. Fourment, T. Balan, and J. L. Chenot. Shape optimal design in forging. In S.-F. Shen and P. R. Dawson, editors, *Proceedings of the 5th International Conference On Numerical Methods in Industrial Forming Processes*, pages 557–562, June 1995.

[11] W. J. Gordon and C. A. Hall. Construction of curvilinear coordinate systems and applications to mesh generation. *International Journal for Numerical Methods in Engineering*, 7:461–477, 1973.

[12] D. M. Greig. *Optimisation.* Longman Group Limited, first edition, 1980.

[13] A. L. Gurson. Continuum theory of ductile rupture by void nucleation and growth: Part i - yield criteria and flow rules for porous ductile media. *Journal of Engineering Materials and Technology*, 99:2–15, 1977.

[14] M. E. Gurtin. *An Introduction to Continuum Mechanics.* Academic Press, Inc., first edition, 1981.

[15] R. B. Haber, D. A. Tortorelli, C. A. Vidal, and D. G. Phelan. Design sensitivity analysis of nonlinear structures i: Large-deformation hyperelastic and history-dependent material response. In M. P. Kamat, editor, *Structural Optimization: Status and Promise. Progress in Astronautics and Aeronautics*, volume 150, pages 369–405, 1992.

[16] R. T. Haftka and Z. Gurdal. *Elements of Structural Optimization.* Kluwer Academic Publishers, third edition, 1992.

[17] C. S. Han, R. V. Grandhi, and R. Srinivasan. Optimum design of forging die shapes using nonlinear finite element analysis. *AIAA Journal*, 31:774–781, 1993.

[18] E. J. Haug, K. K. Choi, and V. Komkov. *Design Sensitivity Analysis of Structural Systems.* Academic Press, Inc., first edition, 1986.

[19] S. R. Kalidindi, C. A. Bronkhorst, and L. Anand. Crystallographic texture evolution in bulk deformation processing of fcc metals. *Journal of the Mechanics and Physics of Solids*, 40:537–569, 1992.

[20] M Kleiber. Shape and non-shape structural sensitivity analysis for problems with any material and kinematic non-linearity. *Computer Methods in Applied Mechanics and Engineering*, 108:73–97, 1993.

[21] S. Kobayashi, S.-I. Oh, and T. Altan. *Metal Forming and the Finite Element Method.* Oxford University Press, first edition, 1989.

[22] A. M. Lush, G. Weber, and L. Anand. An implicit time integration procedure for a set of internal variable constitutive equations for isotropic elasto-viscoplasticity. *International Journal of Plasticity*, 5:521–549, 1989.

[23] D. L. McDowell and J. C. Moosbrugger. Continuum slip foundations of elasto-viscoplasticity. *Acta Mechanica*, 93:73–87, 1992.

[24] M. P. Miller and P. R. Dawson. Reflecting microstructural evolution in hardening models for polycrystalline metals. In S.-F. Shen and P. R. Dawson, editors, *Proceedings of the 5th International Conference On Numerical Methods in Industrial Forming Processes*, June 1995.

[25] S. Mukherjee and Q. Zhang. Design sensitivity in problems involving material and geometric nonlinearities. *International Journal of Solids and Structures*, 31:1793–1827, 1994.

[26] W. Prager and P.G. Hodge. *Theory of Perfectly Plastic Solids.* John Wiley & Sons, Inc, first edition, 1951.

[27] J. C. Simo and R. L. Taylor. Consistent tangent operators for rate-independent elastoplasticity. *Computer Methods in Applied Mechanics and Engineering*, 48:101–118, 1985.

[28] J. Sokolowski and J.-P. Zolesio. *Introduction to Shape Optimization - Shape Sensitivity Analysis.* Springer-Verlag, first edition, 1992.

[29] S. P. Timoshenko and J. N. Goodier. *Theory of Elasticity.* McGraw-Hill Book Company, third edition, 1970.

[30] J. J. Tsay and J. S. Arora. Nonlinear structural design sensitivity analysis for path dependent problems. part i: General theory. *Computer Methods in Applied Mechanics and Engineering*, 81:183–208, 1990.

[31] J. J. Tsay, J. E. B. Cardoso, and J. S. Arora. Nonlinear structural design sensitivity analysis for path dependent problems. part ii: Analytical examples. *Computer Methods in Applied Mechanics and Engineering*, 81:209–228, 1990.

[32] C. A. Vidal and R. B. Haber. Design sensitivity analysis for rate independent elastoplasticity. *Computer Methods in Applied Mechanics and Engineering*, 107:393–431, 1993.

[33] C. A. Vidal, H.-S. Lee, and R. B. Haber. The consistent tangent operator for design sensitivity analysis of history-dependent response. *Computing Systems in Engineering*, 2:509–523, 1991.

[34] G. G. A. Weber. *Computational Procedures For a New Class of Finite Deformation Elastic-Plastic Constitutive Relations.* PhD thesis, Massachusetts Institute of Technology, 1988.

[35] Y. Y. Zhu and T. Zacharia. Application of damage models in metal forming. In S.-F. Shen and P. R. Dawson, editors, *Proceedings of the 5th International Conference On Numerical Methods in Industrial Forming Processes*, June 1995.

Appendix

In this appendix, we derive the weak form of the governing equations for both the direct and the sensitivity problems that define the large deformation of an elasto-viscoplastic material. This form will be used with the finite element version of PDESOLVE which is currently being developed.

The weak form of the governing equations for the direct problem is obtained by taking the weighted average residual of the eqs.(10)-(12) over the volume V_N (updated Lagrangian description) to give

$$\int_{V_N} \mathbf{P}:\nabla\hat{\mathbf{u}}\, dV = \int_{V_N} \rho_N \mathbf{b}\cdot\hat{\mathbf{u}}\, dV + \int_{A_N^\sigma} \bar{\mathbf{t}}\cdot\hat{\mathbf{u}}\, dA \tag{95}$$

where $\hat{\mathbf{u}}$ is a weighting function. To obtain the weak form of the sensitivity equations, we first write eq.(35) referred to B_0 (total Lagrangian description) as

$$\nabla_0\cdot\delta\mathbf{P} - \nabla_0\cdot(\mathbf{P}\cdot\delta\mathbf{L}_0^T) + \mathbf{P}\cdot(\nabla_0\cdot\delta\mathbf{L}_0^T) + \delta\rho_0\mathbf{b} = 0 \tag{96}$$

We note here that the derivation of this equation makes use of the reference domain B_R shown in fig.2. As mentioned before, this domain is typically used when dealing with preform shape design problems. The weighted averaged residual of eq.(96) can be written as

$$\int_{V_0} (\delta\mathbf{P} - \mathbf{P}\cdot\delta\mathbf{L}_0^{\mathrm{T}}):\nabla_0\hat{\mathbf{u}}\, dV = \int_{V_0} \delta\rho_0\mathbf{b}\cdot\hat{\mathbf{u}}\, dV + \int_{V_0} [\mathbf{P}\cdot(\nabla\cdot\delta\mathbf{L}_0^{\mathrm{T}})]\cdot\hat{\mathbf{u}}\, dV + \int_{A_0^\sigma} \left[(\delta\mathbf{P} - \mathbf{P}\cdot\delta\mathbf{L}_0^T)\cdot\mathbf{n}_0\right]\cdot\hat{\mathbf{u}}\, dA \tag{97}$$

The last term of this equation is equivalent to

$$\int_{A^\sigma} \left[\delta\bar{\mathbf{t}} - \mathbf{F}_u\cdot\delta\mathbf{L}_0\cdot\mathbf{F}_u^{-1}:(\mathbf{n}\otimes\mathbf{n})\bar{\mathbf{t}} + (\delta\mathbf{F}_u\cdot\mathbf{F}_u^{-1}:(\mathbf{I}-\mathbf{n}\otimes\mathbf{n}))\bar{\mathbf{t}}\right]\cdot\hat{\mathbf{u}}\, dA \tag{98}$$

which is derived using the traction boundary condition $\bar{\mathbf{t}} = \boldsymbol{\sigma}\cdot\mathbf{n}$, its variation $\delta\bar{\mathbf{t}} = (\delta\boldsymbol{\sigma} - \boldsymbol{\sigma}\cdot\delta\mathbf{L}^{\mathrm{T}})\cdot\mathbf{n} + (\mathbf{n}\cdot\delta\mathbf{L}^T\cdot\mathbf{n})\bar{\mathbf{t}}$ and the relations (see fig. 2) $\delta\mathbf{L} = \delta\mathbf{F}_T\cdot\mathbf{F}_T^{-1}$ $\mathbf{F}_T = \mathbf{F}_u\cdot\mathbf{F}_R$, $\delta\mathbf{L}_0 = \delta\mathbf{F}_R\cdot\mathbf{F}_R^{-1}$, $\mathbf{P} = det(\mathbf{F}_u)\boldsymbol{\sigma}\cdot\mathbf{F}_u^{-T}$, and $\mathbf{n}dA = det(\mathbf{F}_u)\mathbf{F}_u^{-T}\cdot\mathbf{n}dA_0$. Therefore, the weak form of the sensitivity equations can be written as

$$\begin{aligned}\int_{V_0} (\delta\mathbf{P} - \mathbf{P}\cdot\delta\mathbf{L}_0^{\mathrm{T}}):\nabla\hat{\mathbf{u}}\, dV = \int_{V_0} \delta\rho_0\mathbf{b}\cdot\hat{\mathbf{u}}\, dV + \int_{V_0} [\mathbf{P}\cdot(\nabla\cdot\delta\mathbf{L}_0^{\mathrm{T}})]\cdot\hat{\mathbf{u}}\, dV + \\ \int_{A^\sigma} \left[\delta\bar{\mathbf{t}} - \mathbf{F}_u\cdot\delta\mathbf{L}_0\cdot\mathbf{F}_u^{-1}:(\mathbf{n}\otimes\mathbf{n})\bar{\mathbf{t}} + (\delta\mathbf{F}_u\cdot\mathbf{F}_u^{-1}:(\mathbf{I}-\mathbf{n}\otimes\mathbf{n}))\bar{\mathbf{t}}\right]\cdot\hat{\mathbf{u}}\, dA\end{aligned} \tag{99}$$

If the reference configuration B_R coincides with the undeformed configuration of the body B_0, then $\mathbf{F}_R = \mathbf{I}$ and, hence, $\delta\mathbf{L}_0 = \mathbf{0}$. With this value, eq.(97) reduces to

$$\int_{V_0} \delta\mathbf{P}:\nabla\hat{\mathbf{u}}\, dV = \int_{V_0} \delta\rho_0\mathbf{b}\cdot\hat{\mathbf{u}}\, dV + \int_{A^\sigma} \left[\delta\bar{\mathbf{t}} + (\delta\mathbf{F}_u\cdot\mathbf{F}_u^{-1}:(\mathbf{I}-\mathbf{n}\otimes\mathbf{n}))\bar{\mathbf{t}}\right]\cdot\hat{\mathbf{u}}\, dA \tag{100}$$

Equation (100) is mainly used when dealing with die design problems while keeping the preform shape fixed.

Discretization of the weak form for the direct problem, eq.(95), using standard finite element techniques leads to a set of nonlinear algebraic equations for the nodal values of $\mathbf{u}$ which is solved using the Newton method. On the other hand, discretization of the weak form for the sensitivity problem, eq.(99) or eq.(100), produces a system of linear algebraic equations in the nodal values of $\delta\mathbf{u}$ which are solved at each time step for each design variable once the solution of the direct problem is obtained. At the end of the specified time interval, the computed solution for the field variables and their variations are input to the optimizer in order to obtain the new values of the design parameters. This procedure is then repeated until values of the design parameters that minimize the objective function within a prescribed tolerance are found.